THÈSES

PRÉSENTÉES

A LA FACULTÉ DES SCIENCES
DE L'UNIVERSITÉ DE PARIS

POUR OBTENIR

LE GRADE DE DOCTEUR ES SCIENCES MATHÉMATIQUES

PAR

Jacques PITRAT

1re THESE. —

"Réalisation de programmes de démonstration de théorèmes utilisant des méthodes heuristiques"

2e THESE. —

PROPOSITIONS DONNÉES PAR LA FACULTÉ.

"Approximation de fonctions expérimentales en présence de bruit"

Soutenues le 28 Avril 1966 devant la Commission d'examen.

MM. DE POSSEL *Président.*

VILLE

ARSAC } *Examinateurs.*

THÈSES

PRÉSENTÉES

A LA FACULTÉ DES SCIENCES
DE L'UNIVERSITÉ DE PARIS

POUR OBTENIR

Le Grade DE DOCTEUR ES SCIENCES MATHEMATIQUES

PAR

Jacques PITRAT

1re **THESE.** —

"Réalisation de programmes de démonstration de théorèmes utilisant des méthodes heuristiques"

2e **THESE.** —

PROPOSITIONS DONNÉES PAR LA FACULTÉ.

"Approximation de fonctions expérimentales en présence de bruit"

Soutenues le 28 Avril 1966 **devant la Commission d'examen.**

MM. DE POSSEL *Président.*

VILLE

ARSAC } *Examinateurs.*

UNIVERSITEITSBIBLIOTHEEK - LEIDEN -

FACULTE DES SCIENCES DE PARIS

Janvier 1966

PROFESSEURS

Le Doyen . . . . . M. ZAMANSKY

PRENANT	T	Anatomie et histologie comparée
GRASSE	T	Evolution des êtres comparée
PREVOST	T	Chimie organique
WYART	T	Minéralogie & cristallographie
TEISSIER	T	Zoologie
AUGER	T	Physique quantique & relativité
MONNIER	T	Physiologie générale
PIVETEAU	T	Paléontologie
ROCARD	T	Physique E.N.S.
CARTAN	T	Mathématiques E.N.S.
LAFFITE	T	Chimie générale
N.....	T	Géométrie supérieure
COULONB	T	Physique du globe
Mlle COUSIN	T	Biologie animale SPCN
CHRETIEN	T	Chimie minérale
BOCQUET	T	Zoologie
KADTLER	T	Physique E.N.S.
EPHRUSSI	T	Génétique
GAUTHERET	T	Biologie végétale PCB
LUCAS R.		Recherches physiques
THOMAS A.	T	Biologie cellulaire
MORAND	T	Physique enseignement
SOLEILLET	T	Physique P.C.B.
FORTIER	T	Mécanique expérimentale des fluides
DRACH	T	Zoologie
QUENEY	T	Météorologie & dynamique atmosphérique
GALLIEN	T	Embryologie
EICHHORN	T	Botanique
DE CUGNAC	T	Biologie végétale SPCN
Mlle CAUCHOIS	T	Chimie physique
THELLIER	T	Physique du globe
PONCIN	T	Mécanique générale
DUBREIL	T	Arithmétique & théorie des nombres
QUELET	T	Chimie organique
CAGNIARD	T	Géophysique appliquée
CHAMPETIER	T	Chimie macromoléculaire
CUVILLIER	T	Micropaléontologie
JUNG	T	Géologie
TRILLAT	T	Microscopie & Diffract. Electronique
WIEMANN	T	Chimie organique & structurale
VASSY	T	Physique de l'atmosphère
DESTOUCHES	T	Théorie physique
AMIEL	T	Chimie générale
HOCART	T	Minérallogie & cristallographie
J.P. MATHIEU	T	Physique optique
COUTEAUX	T	Cytologie
CHOQUET	T	Théorie des fonctions & topologie
FELDMANN	T	Biologie végétale marine
JOST	T	Physiologie comparée

FORTET	T	Calcul des probabilités & physique mathématiques
SCHWARTZ	T	Calcul différentiel & intégral
CHOUARD	T	Physiologie végétale
MALAVARD	T	Aviation (technique aéronautique)
BRELOT	T	Analyse supérieure
NORMANT	T	Synthèse organique
BENARD	T	Chimie minérale
DUGUE	T	Statistique
SOULAIRAC	T	Psychophysiologie
ULRICH	T	Physiologie végétale appliquée
KIRRMANN	T	Théories chimiques
CHADEFAUD	T	Botanique
Mlle LE BRETON	T	Physiologie de la nutrition
LEGONG	T	Application de l'analyse à la géométrie
DEVILLERS	T	Anatomie & histologie comparées
EHRESMANN	T	Topologie algébrique
FRANCON	T	Physique SPCN
GLANGEAUD	T	Géographie physique, géologie dynamique
DODEMENT	T	Mathématique MGP
PISOT	T	Technique mathématique de la physique
ROCH	T	Géologie
SCHATZMANN	T	Astrophysique
TERMIER	T	Géologie générale
ZAMANSKY	T	Mathématiques générales
LENNUIER	T	Physique M.G.P.
ROUTHIER	T	Géologie appliquée
Me TONNELAT	T	Physique théorique
DIXMIER	T	Mathématiques M.P.C.
SOUCHAY	T	Chimie générale
AIGRAIN	T	Electrotechnique
BRUSSET	T	Chimie systématique
M. LEVY	T	Physique théorique des hautes énergies
Mme CHAIX	T	Chimie biologique
Me HUREL PY	T	Biologie végétale PCB
PANNETIER	T	Chimie générale
BRUN	T	Mécanique des fluides I
Me DUBREIL	T	Mathématiques (Agrégation)
Me LELONG	T	Mathématiques II E,N.S.
BELLAIR	T	Géologie S.P.C.N.
COTE	T	Physique électricité
DUBOIS J.E.	T	Chimie organique
LAMOTTE	T	Zoologie E.N.S.
OLMER	T	Energétique générale
GAUTHIER	T	Mécanique appliquée
BARCHEWITZ	T	Chimie physique
BROSSEL	T	Physique atomique
BUSER	T	Physiologie comparée
CAMUS	T	Physiologie végétale
CURIEN	T	Minéralogie & cristallographie
Me CHARNIAUX COTTON	T	Evolution des être organisés
POSSOMPES	T	Zoologie
PULLMANN	T	Chimie quantique
TEILLAC	T	Physique nucléaire & radioactivité
VILLE	T	Econométrie
WILLEMART	T	Chimie PCB
FREYMANN	T	Recherches physiques
ROLLET	T	Chimie minérale
Mlle JOSIEN	T	Chimie S.P.C.N.
CHEVALLEY	T	Géométrie algébrique & théorie des groupes

GERMAIN	T	Mécanique théorique des fluides
LUCAS G.	T	Géologie
ALLARD	T	Chimie physique
BERTHELOT	T	Physique des particules fondamentales
BRICARD	T	Météorologie
Mme ALBE-FESSARD	T	Psycho-physiologie
M. JULIA	T	Etude des molécules naturelles complexes
MAGAT	T	Physioco-chimie des rayonnements
Mlle QUINTIN	T	Electrochimie
MONOD	T	Chimie du métabolisme
DE POSSEL	T	Analyse numérique
CHARLOT	T	Chimie analytique
CHATELET	T	Chimie PCB
MATTLER	T	Physique SPCN
BOUISSIERES	T	Radiochimie
Mme COUTURE	T	Thermodynamique & Mécanique physique
FRANC	T	Biologie animale PCB
SCHNELL		Botanique tropicale
STOLKOWSKI	T	Physiologie générale
ACHER	T	Chimie biologique
BAUDOIN	T	Biologie animale PCB
BEAUMONT		Biologie animale SPCN
BLAMONT		Géophysique ionosphère
BOUREAU	T	Botanique
CURIE D.		Physique M.G.P.
DURAND DELGA		Géologie
HELLER	T	Physiologie végétale
TORTRAT	T	Calcul des probabilités
CAILLEUX	T	Géologie S.P.C.N.
MAGNAN		Physique électricité
DAUDEL	T	Mécanique ondulatoire
Mlle FOURCROY		Biologie végétale S.P.C.N.
Me CHOQUET BRUHAT		Mécanique analytique & mécanique céleste
Mlle VEIL	T	Physiologie générale
LWOFF	T	Microbiologie
SIESTRUNCK	T	Mécanique physique & expérimentale
CABANNES	T	Mécanique générale
CAIRE	T	Géographie physique & géologie dynamique
COMOLET	T	Mécanique des fluides
DELLOUE	T	Physique de l'exosphère
HOUILLON		Biologie animale PCB
OLIVIER		Antropologie
SAMUEL	T	Mathématiques (ENS Jeunes filles)
THIRY	T	Mécanique céleste
ABELES	T	Physique S.P.C.N.
AMAT	T	Thermodynamique
AUBOIN	T	Géologie
Me BENOIT-GUEUTAL	T	Physique nucléaire & radio-activité
BOUGIS		Biologie marine Villefranche sur mer
BRUHAT	T	Mathématique MPC
DEHEUVELS	T	Mathématiques propédeutique
FURON	T	Géologie S.P.C.N.
Mme GANS	T	Génétique
GENEVES		Biologie végétale P.C.B.
GUILBERT		Physique CUS Orléans
HUPE	T	Pétrographie
HOLLANDE	T	Biologie générale

LIONS	T	Analyse numérique
MAITTE	T	Chimie organique
NICOLAS		Géologie appliquée
PANIGEL		Biologie générale
RAVIER		Pétrographie
ROSEAU	T	Mécanique théorique des fluides
ROUBINE		Electronique
SILBER	T	Chimie P.C.B.
UEVERSFELD	T	Electronique
YVON	T	Théories physiques
HERVE	T	Mathématiques
Me PECKER-WIMEL	T	Astrophysique
BALKANSKI		Physique M.P.C.
LE CORRE	T	Electricité
NATAF	T	Physique nucléaire & radioactivité
ECHALIER		Biologie animale P.C.B.
BARRAUD		Physiologie cellulaire
DARS		Géologie structurale & appliquée
CAMEFORT	T	Botanique (agrégation)
VICHNIEWSKY	T	Mécanique industrielle
BLAQUIERE		Electronique
ARNOUS	T	Physique théorique
TATIBOUET		Chimie (CSU Orléans)
MALLIAVIN	T	Mathématiques propédeutique
GAUDEMAR		Chimie propédeutique
BROUSSE	T	Mécanique théorique
ROSCH	T	Astronomie
GUY	T	Physique expérimentale
LAZARD		Mathématique MPC
MOREL P.		Physique MPC
ARSAC		Programmation

LEFEBVRE		Chimie MPC
CHAPEVILLE		Biochimie
BENOIT		Physique CPEM
RIO		Chimie CPEM
JAUZEIN		Géologie ENS
PREVOST G.		Biologie CPEM
FAVARD P.		Biologie CPEM
Mlle FICINI		Chimie CPEM
LEQUEUX		Astronomie
COHEN TANNOUDJI		Physique atomique
TAVERNIER		Physique CPEM
DURAND M.		Biologie animale SPCN
PASCAUD		Physiologie
Mme PETIT		Biologie animale CPEN
CAGNAC		Physique agrégation
FALK-VAIRANT		Physique nucléaire
RIGAUDY		Chimie SPCN
SIGWALT		Chimie SPCN
FOURNET		Electricité générale
COLLONGUES		Chimie CPEN
DUBRET		Biochimie
MOREL F.		Physiologie comparée
TALBOT		Génétique chimique
Me NOUGAREDE		Botanique
FRANCOIS		Zoologie
IVANOFF		Océanographie physique
CADIOT		Chimie organique
BARUCH		Physique propédeutique
NOZIERES		Physique générale
BOK		Physique propédeutique
SLONIMSKI		Génétique
KRASNER	T	Mathématiques
BENZECRI	T	Statistiques

VINH MAU	Physique MGP
DUQUESNES	Physique CPEM
SCHUTZENBERGER	Programmation
TREMILLON	Chimie MPC
FAYARD	Chimie MPC
BUVET	Chimie CPEM
AUTHIER	Physique MPC
PICINBONO T	Physique CPEM
BONNEFILLE	Electrotechnique
Me PUISEUX DAO	Biologie végétale

Le Secrétaire Général

R. POULLAIN

Digitized by the Internet Archive
in 2026

https://archive.org/details/IA41546023_0032

UNIVERSITE DE PARIS

FACULTE DES SCIENCES D'ORSAY

Janvier 1966

PROFESSEURS

MANGENOT G.	T	Biologie végétale
JACQUINOT P.	T	Optique
GUINIER A.	T	Electricité & physique des solides
GRIVET P.	T	Electronique
BLANC-LAPIERRE A.	T	Accélérateur linéaire
L'HERITIER P.	T	Génétique des virus
LEMEE G.	T	Ecologie végétale
MOYSE A.	T	Physiologie végétale
DODE M.	T	Chimie thermodynamique
DENY J.	T	Mathématiques
RIVIERE A.	T	Géologie générale & sédimentologie
MICHEL A.	T	Chimie minérale
MAY R.	T	Anatomie comparée
DELANGE H.	T	Mathématiques
CHAPELLE J.	T	Physique expérimentale
LACOMBE P.	T	Métallurgie
BENOIT P.	T	Physiologie générale
GUINOCHET J.	T	Biologie végétale SPCN
NERON A.	T	Mathématiques
NOZERAN R.	T	Botanique approfondie
CASTAING R.	T	Thermodynamique
ELLENBERGER F.	T	Géologie structurale
LEDERER E.	T	Chimie biologique
RIZET J.	T	Génétique
LENDER T.	T	Biologie animale SPCN
POITOU G.	T	Mathématiques
FREON P.	T	Chimie organique
TONNELAT	T	Biologie physico-chimique
BRUNN Jan	T	Géologie nucléaire
JEAN M.	T	Physique nucléaire
BERGERARD J.	T	Zoologie
ROGER J.	T	Paléontologie
LEFORT M.	T	Chimie nucléaire
LAGARRIGUE	T	Physique des hautes énergies
JULLIEN J.	T	Chimie structurale organique
ARNULF A.	T	Optique
ROUAULT M.	T	Diffraction électronique
MARECHAL A.	T	Optique théorique et appliquée
MAZET R.	T	Mécanique générale
DAVID S.	T	Chimie agrégation
GUERIN H.	T	Chimie des gaz et des combustibles
LESIEUR L.	T	Mathématiques MPC et algèbre
KOURGANOFF	T	Astronomie
ARNOULT P.	T	Physique électronique
KAHANE J.P	T	Mathématiques
BROCHARD J.	T	Physique propédeutique
Mme ROIZEN	T	Physique CPEM
BERTEIN F.	T	Electronique
MALGRANGE B.	T	Mathématiques
CORABOEUF E.	T	Physiologie animale

FRIEDEL J.	T	Physique des solides
CHEVAUGEON	T	Cryptogamie
MEYER P.	T	Physique théorique & atomique
LEHMANN P.	T	Accélérateur linéaire
CHABRAL R.	T	Physique propédeutique
DELCROIX J.M.	T	Physique des plasmas
CERF J.	T	Mathématiques
D'ESPAGNAT	T	Physique théorique
SCHAEFFER P.	T	Microbiologie
DE GENNES P.	T	Physique des solides
Mme MARTY N.		Physique propédeutique
GOURDIN M.		Physique théorique
PEREZ-Y-JORBA		Physique propédeutique
GORENFLOT E.		Biologie végétale SPCN
RIOU M.		Physique nucléaire
BROUSSE R.		Géologie SPCN
LIORET C.		Physiologie végétale
CLAUSER H.		Chimie biologique
JANCOVICI B.		Physique atomique
LURCAT J.		Physique propédeutique

LE BERRE J.	Entomologie
LAVOREL J.	Biophysique
PERIO M.	Physique propédeutique
FETIZON M.	Stéréochimie
CHEMLA M.	Chimie des isotopes
MAZIERES C.	Chimie SPCN
VILKAS M.	Chimie CPEM
BLANDIN A.	Physique des solides
BRUN G.	Génétique
MERCOUROFF	Physique agrégation
ANDRE J.	Microscopie électronique
CHEVALLIER F.	Physiologie animale
Mlle MARCOU	Biologie cellulaire CPEM
BOURDU R.	Biologie végétale
BERKALOFF A.	Biologie animale CPEM
Mlle LEFORT M.	Biologie cellulaire CPEM
Mme LAMBERT	Physique CPEM
BIZOT J.C	Physique MPC
GUIRAUD J.P	Mécanique
LABOUESSE B.	Chimie SPCN

Je tiens à exprimer ma gratitude à Monsieur VILLE pour sa bienveillance à mon égard et ses encouragements constants.

Monsieur DE POSSEL à bien voulu s'intéresser à mon travail et accepter de présider le jury de cette thèse. Je lui en suis très reconnaissant. Je remercie également Monsieur ARSAC qui m'a proposé un second sujet et m'a guidé dans sa préparation.

J'exprime ma reconnaissance au fond de recherche IBM qui a mis à ma disposition une heure d'IBM 7090 et à l'Institut Blaise Pascal du C.N.R.S. qui m'a permis de faire un nombre très important d'heures sur son IBM 704. Sans eux cette thèse n'aurait pu être réalisée.

Enfin je remercie la Direction Technique des Armements Terrestres qui m'a autorisé à poursuivre ces recherches et a bien voulu m'aider pour la réalisation matérielle de ce travail.

TABLE DES MATIERES

Pour démontrer un théorème, il faut savoir ce qu'on a le droit de faire : c'est ce qu'étudie le chapitre 2. Il faut également choisir parmi ce que l'on peut faire : on utilise pour cela des heuristiques On étudie ces méthodes au cours du chapitre 3. Le chapitre 4 étudie quelques questions annexes, le chapitre 5 traite de la réalisation des programmes. Les résultats trouvés par le programme sont donnés dans le chapitre 6.

Les parties marquées d'un astérisque ne sont pas indispensables à la compréhension de l'ensemble et peuvent être omises en première lecture.

1 - POSITION DU PROBLEME

1A - Approche choisie 1

1B - Algorithmes et heuristiques 2

1C - Aspect psychologique 8

2 - ETUDE LOGIQUE

2A - Généralités sur les systèmes formels et notations

2A1 - Description d'un système formel 13

2A2 - Présentation, représentation, interprétation 18

2A3 - Métalangues 19

* 2A4 - Théorèmes de limitation et intelligence artificielle 22

2B - Algorithme de dérivation 25

2C - Etude générale des démonstrations

2C1 - Formules décrivant une démonstration 34

2C2 - Représentation graphique des démonstrations 41

2C3 - Nombres qui caractérisent la difficulté d'une démonstration 49

2D - MMthéorèmes et MMMthéorèmes utilisés par le programme 51

* 2E - Deux MétaMétathéorèmes : les théorèmes de déduction et du substitution

2E1 - Généralités 55

2E2 - Le théorème de déduction 55

2E3 - Le théorème de substitution 65

2E4 - Conclusion 70

3 - ASPECT HEURISTIQUE

3A - Impossibilité de l'approche combinatoire 73

3B - Elimination de résultats.

3B1 - Etude de la force d'un résultat 74

3B2 - Détermination de l'intérêt d'un résultat 82

3B3 - Elimination de résultats par les résultats déjà démontrés 92

3C - Elimination d'essais 99

3D - Influence de l'ordre dans lequel on fait les essais 100

3E - Intérêt de l'utilisation des divers niveaux linguistiques 101

* 4 - QUELQUES PROBLEMES ANNEXES

4A - Les définitions

4A1 - Généralités 105

4A2 - Application d'une définition 106

4A3 - Heuristiques utilisées pour appliquer les définitions 106

4A4 - Conclusion 107

4B - L'ambiguité

4B1 - Généralités 108

4B2 - Ambiguité dûe à l'utilisation des Mthéorèmes 108

4B3 - Ambiguité effective 109

4B4 - Conclusion 112

4C - L'apprentissage

4C1 - Généralités 113

4C2 - Possibilités d'apprentissage du programme 116

4C3 - Conclusion 117

4D - Démonstrations de conjectures

4D1 - L'arbre "ou - et" 117

4D2 - Mise à jour de l'arbre 120

4D3 - Utilisation du procédé 121

* 5 - PROGRAMMATION

5A - Les trois premiers programmes

5A1 - Le premier programme 123

5A2 - Le deuxième programme 123

5A3 - Le troisième programme 125

5B - Utilisation de l'outillage de programmation en listes 126

5C - Le quatrième programme 128

5D - Archivage des résultats dans des arborescences 131

6 - RESULTATS OBTENUS PAR LE PROGRAMME

Axiomatique 1 140

Axiomatique 2 144

Axiomatique 3 150

Axiomatique 4 151

Axiomatique 5 153

Axiomatique 6 155

7 - CONCLUSION

7A - Améliorations souhaitables 161

7B - Principales difficultés rencontrées 162

7C - Bilan de l'étude 163

SYMBOLES UTILISES

Arithmétique :

- opérations

$+$	addition
$-$	soustraction (moins binaire)
$\times$	multiplication
$/$	division
$\uparrow$	exponentiation
$\ominus$	moins (unaire)

quand il n'y a pas d'ambiguité, on utilisera aussi - pour indiquer le moins unaire et l'absence de symbole pour la multiplication (multiplication implicite). Σ désigne la sommation algébrique.

- relations

$=$	égalité
$>$	supérieur
$\geqslant$	supérieur ou égal
$<$	inférieur
$\leqslant$	inférieur ou égal

Suites de symboles

L'opération de concaténation est représentée par un symbole vide : $\alpha\beta$ indique la concaténation des suites α et β. Si α est ab et β est bab, $\alpha\beta$ sera abbab.

Si α est une suite de symboles, α^n représente la concaténation de n fais la suite α. Si α est ab, α^3 est ababab.

Λ désignera la suite de symboles vide.

On définira dans la remarque qui termine 3B1 les relations suivantes qui portent sur des suites de symboles.

$=$

$>$

$<$

$\geqslant$

$\leqslant$

On représente par p/α l'opération de substitution de la suite de symboles α à la variable p.

Opérations liées à la dérivation

Les symboles qui représentent une dérivation sont définis en 2B :

$+$

$\dashv\!\!\rightarrow$

$\dashv\!\!\!\rightarrow$

Opérations de la langue

unaires	$\sim$	négation
	$\Diamond$	possibilité
	N	nécessité
binaires	$\supset$	implication
	$\wedge$	et
	$\vee$	ou inclusif
	$\mid$	incompatibilité (Sheffer's stroke)
	$\equiv$	identité

Opérations de la métalangue ou Mlangue

$\Rightarrow$	implication
,	et
$\Leftrightarrow$	identité

Opérations de la MMMlangue

$\rightsquigarrow$	implication
$\square$	et

Certains symboles ont plusieurs sens :

- 2 sens : $->$, $\geqslant$, $<$, $\leqslant$ qui ont leur sens arithmétique classique et peuvent porter sur des suites de symboles.
  - symbole vide : peut représenter la concaténation ou la multiplication.
  - / peut représenter la division ou les substitutions.
  - + peut être l'addition ou servir à représenter une dérivation.
- 3 sens : = peut représenter l'égalité de formules arithmétiques, de suites de symboles ou une définition.

Cela n'est pas génant : le contexte, qui indique de quelle nature sont les opérandes, lève immédiatement l'ambiguité.

Nous utiliserons les abréviations :

Mlangue pour métalangue

M^nlangue pour métalangue de la $M^{(n-1)}$langue

Mthéorème pour Métathéorème

M^nthéorème pour un théorème de la M^nlangue

Connective étant pris comme synonyme d'opération, une M^nconnective sera une connective de la M^nlangue : par exemple $\square$ est une MMMconnective.

Ceci est explicité en 2A3.

CHAPITRE I

POSITION DU PROBLEME

1A - APPROCHE CHOISIE

Le mathématicien peut être amené à avoir trois types d'activités très différentes :

1 - Il essaye de démontrer ou de réfuter une conjecture qu'on lui a donnée dans une axiomatique qui lui a été également donnée. Il essaye d'établir une dérivation de la formule (ou de sa négation) à partir des axiomes en utilisant les régles de dérivation de l'axiomatique. Cette activité est celle de l'étudiant à qui l'on a donné un problème à résoudre. C'est celle du mathématicien qui essaye de démontrer le théorème de Fermat.

2 - Il peut essayer de voir quels sont les théorèmes intéressants qu'il peut démontrer dans une axiomatique qui lui a été donnée. Il est libre de choisir les théorèmes qu'il va essayer de démontrer. Il est ramené de temps en temps au problème précédent quand il a l'intuition qu'un certain énoncé est un théorème qu'il prend comme conjecture. Ce travail correspond à celui d'un nombre important de mathématiciens, qui essayent de trouver des conséquences intéressantes et nouvelles d'une axiomatique donnée. Pendant longtemps du reste les mathématiciens n'ont connu que ces deux activités, car ils considéraient que les axiomes avaient un caractère nécessaire et ils n'avaient pas vu le caractère conventionnel des axiomatiques. Aussi ils ont essayé de trouver de nouveaux théorèmes en arithmétique, en algèbre, en géométrie, en analyse sans essayer de vouloir changer l'axiomatique sur laquelle ils s'appuyaient.

3 - Il peut essayer de définir une nouvelle axiomatique. Les buts de cette recherche peuvent être multiples : on peut vouloir exprimer une théorie déjà connue à partir d'axiomes moins nombreux ou d'une interprétation plus intuitive. On peut aussi vouloir formaliser des structures qui ont un intérêt esthétique ou pour une application particulière. On peut enfin vouloir faire une synthèse de plusieurs théories.

Ceux qui jusqu'à présent ont essayé de faire faire par une machine le travail du mathématicien se sont tous tournés vers le premier type d'activités. Les programmes étaient limités à une seule axiomatique et on leur donnait des conjectures qu'ils essayaient de démontrer, souvent avec succès.

Le troisième type d'activité est certainement le plus noble, mais il est extraordinairement compliqué à simuler, car les critères qui déterminent le choix d'une nouvelle axiomatique sont très complexes et résultent souvent du travail de plusieurs générations qui ont réussi à dégager que certains concepts avaient une grande importance pratique.

Faire faire ceci par une machine semble totalement hors de portée de nos connaissances actuelles, c'est pourquoi cette étude essaye de simuler le deuxième type d'activités :

Le but est d'écrire un programme qui a comme donnée une axiomatique (ce qui lui permet de traiter diverses axiomatiques). A partir de cette axiomatique, il va essayer de démontrer des théorèmes "intéressants", sans que l'énoncé de ces théorèmes lui soit donné : il n'a aucune information sur ce qu'il doit trouver. La difficulté de l'étude tient essentiellement dans ce fait : il n'y a aucune difficulté à faire démontrer des théorèmes par un automate, il suffit de les énumérer. Mais par ce procédé on a une multitude de résultats sans intérêt. Il faut d'une part définir un critère d'intérêt qui coïncide avec notre intuition pour ne garder que ceux qui satisfont à ce critère et d'autre part faire en sorte que le programme fasse seulement les essais qui ont une chance raisonnable de conduire à un résultat intéressant.

En résumé le problème va être double :

- savoir reconnaître si un résultat est intéressant.

- faire d'abord les essais qui ont une grande probabilité de donner un résultat intéressant.

Si l'on n'a résolu que la première difficulté on aura peu de résultats, si l'on n'a résolu que la deuxième, on aura beaucoup de résultats, dont certains, intéressants, seront noyés dans une masse de résultats totalement dénués d'intérêt.

Le programme a en plus la possibilité d'interrompre son étude de l'axiomatique pour essayer de démontrer une conjecture qu'on lui aura donnée. Il se peut du reste fort bien qu'il l'ait déjà démontrée au moment où on la lui pose. Il peut ainsi également avoir le premier type d'activités, mais ce n'est pas le but essentiel de cette étude.

Pour comprendre ce qui suit, il faut bien voir que le but que l'on s'est donné n'est pas de démontrer des théorèmes nouveaux, mais d'essayer d'établir des mécanismes qui permettent de démontrer des théorèmes nouveaux. Partant sans aucune information à ce sujet (il n'y a jamais rien eu de publié sur cette approche du problème) et ne disposant que de machines ayant une mémoire limitée (10^6 bits est dérisoire vis à vis de la capacité de la mémoire d'un mathématicien), on s'est attaqué à des axiomatiques très simples et en particulier à des axiomatiques de la logique des propositions. Il est bien évident que l'on a peu de chance d'y découvrir un théorème nouveau (mais le programme a trouvé des démonstrations probablement nouvelles, il est difficile de recenser tout ce qui a été fait dans ce domaine, et à coup sur inconnues de celui qui a fait le programme).

On aurait couru à un échec certain en étudiant dès le début des axiomatiques très compliquées. Les axiomatiques essayées offrent du reste de réelles difficultés pour le mathématicien, et pour en juger il suffit d'essayer par exemple de voir ce que l'on trouve dans une axiomatique difficile, comme l'axiomatique 3, et de le comparer avec ce que le programme a trouvé.

Mais, répétons-le, **on n'a pas voulu tellement obtenir des résultats nouveaux qu'essayer de trouver des mécanismes qui permettent d'obtenir des résultats nouveaux.**

Les axiomatiques étudiées ont, il faut bien le reconnaitre, peu d'applications pratiques, mais d'approfondir un problème simple peut permettre par la suite d'aborder des problèmes plus complexes qui, eux, auront une importance pratique.

1B - ALGORITHMES ET HEURISTIQUES

Prenons d'abord pour éclaircir la question le premier type d'activités : on a une conjecture que l'on veut démontrer. La solution idéale serait d'avoir un procédé systématique qui indique quand il a terminé, en un temps fini, que la conjecture est dérivable, c'est à dire qu'elle est un théorème, ou qu'elle n'est pas dérivable à partir des axiomes. Nous appellerons une telle procédure un algorithme de démonstration. La logique des propositions admet un tel algorithme : c'est la méthode des tables de vérité justifiée par le théorème de la vérité. En un nombre d'étapes fini, qui peut être très grand si la formule comporte de nombreuses variables, on peut savoir si une formule de la logique des propositions est ou non dérivable. Il existe aussi un algorithme pour la géométrie.

Mais le théorème de Church a ruiné l'espoir que pour toute théorie il existait un algorithme en montrant qu'il ne pouvait exister d'algorithme* pour le calcul des prédicats. Cela est dû à ce que

* Il faut bien voir que cette limitation est valable pour l'homme aussi bien que pour l'automate. Le fait qu'il n'y ait pas d'algorithme n'entraine pas que la machine soit impuissante. Cela entraine simplement qu'elle devra utiliser des méthodes analogues à celles que le mathematicien emploie en pareil cas : les méthodes heuristiques. Ce point important sera développé plus en détail en 2 A 4.

l'ensemble des théorèmes est certes récursivement dénombrable, c'est à dire que l'on peut énumérer tous les théorèmes, mais qu'il n'est pas récursif, car l'ensemble des énoncés qui ne sont pas des théorèmes n'est pas récursivement dénombrable. On peut créer un procédé systématique qui s'arrêtera surement si la conjecture est un théorème ; mais si le procédé ne s'arrête pas, on ne sait pas si on ne l'a pas fait marcher assez longtemps et alors la conjecture serait un théorème ou si la conjecture n'est pas un théorème auquel cas le procédé peut très bien ne jamais s'arrêter.

Les logiciens n'ont pas désespéré pour autant et même pour des axiomatiques indécidables, (c'est à dire pour lesquelles il n'existe pas d'algorithme qui permette, pour toute formule, de dire si elle est ou n'est pas un théorème), ils ont réussi à trouver des algorithmes valables pour des sous-ensembles d'énoncés. Le problème est évidemment de prendre des sous-ensembles aussi vastes que possibles, tout en étant sûr que l'on n'arrivera jamais à trouver un algorithme valable pour tous les énoncés. On appellera par la suite ces procédés des algorithmes partiels.

Le programmation d'un algorithme pose peu de problèmes et ces méthodes font davantage appel à la logique qu'aux machines à calculer qui n'interviennent qu'après une étude mathématique délicate et demandent des programmes relativement courts et faciles à programmer.

On peut citer comme travaux effectués dans cette branche ceux de :

- GILMORE,
- PRAWITZ, PRAWITZ and VOGHERA
- DAVIS, LOGEMANN, LOVELAND, PUTNAM
- HAO WANG.

Le pseudo-algorithme utilisé par Hao Wang est un algorithme pour le calcul des propositions qui indique si sont ou non dérivables (en donnant même un contre exemple dans ce dernier cas) tous les énoncés de la forme :

$$(x_1)(x_2)..(x_n)(Ey_1)..(Ey_n)P(x_1,..x_m,y_1,..y_n.)$$

(en réalité c'est même un peu plus restreint, il y a des contraintes supplémentaires).

La méthode programmée sur IBM 704 a résolu tous les théorèmes de 9 chapitres des Principia. (soit 200 théorèmes du calcul des propositions et 150 du calcul des prédicats) en 8 minutes.

Le problème essentiel dans le cas des algorithmes est qu'ils convergent rapidement. On est évidemment sûr que la machine s'arrêtera et donnera la réponse, mais dans certains cas, cela risque de prendre des milliards d'années.

D'autre part, il faut faire une étude mathématique délicate pour établir un algorithme ou un algorithme partiel.

Dans le cas où l'on n'a qu'un algorithme partiel, le champ d'application de cet algorithme, c'est à dire l'ensemble des formules pour lesquelles il peut dire si elles sont ou non un théorème, peut être trop restreint et ne pas comprendre les énoncés auxquels on s'intéresse.

Enfin on a besoin d'établir d'abord un certain nombre de théorèmes de l'axiomatique pour justifier l'algorithme.

Pour ces quatre raisons il est essentiel dans certains cas d'utiliser des méthodes très diffétes et beaucoup plus proches de celles qu'emploie le mathématicien : ce sont les méthodes heuristiques (ou heuristiques).

Utiliser des heuristiques, c'est utiliser des règles qui ne conduisent pas forcément à la meilleure solution, mais en général en rapprochent (mais parfois en éloignent). La justification d'une

heuristique est essentiellement probabiliste : elle sera utilisée si, dans la majorité des cas, elle donne un résultat intéressant, mais il est fort possible que dans certains cas elle se révèle sans intérêt et même dans d'autres elle peut être néfaste.

On n'a aucune certitude d'arriver à la solution en utilisant des heuristiques, mais on a une grande probabilité d'y arriver si les heuristiques dont on dispose sont bonnes.

Cela parait choquant à première vue pour le mathématicien habitué à la rigueur et au systématique. Mais cela joue un rôle fondamental au cours des recherches qu'il fait. Cela n'apparait pas dans les livres où le mathématicien, une fois arrivé à la solution, donne une démonstration souvent fort éloignée de la démonstration qu'il a d'abord trouvée. C'est malheureux car, d'une part le lecteur a un sentiment d'incapacité en ne voyant pas comment il pourrait trouver une telle démonstration et d'autre part l'étude des méthodes heuristiques utilisées pourrait aider le chercheur qui travaille dans un domaine voisin et surtout pourrait donner des matériaux pour faire une étude scientifique des méthodes heuristiques.

Pratiquement on ne trouve des renseignements à ce sujet que dans des manuels à but pédagogique, les plus remarquables étant les livres de Polya, et en particulier "How to solve it" qui contiennent un grand nombre d'heuristiques, étudiées de façon systématique et surtout axées vers l'enseignement secondaire.

Certains heuristiques ont un caractère général, d'autres sont adaptées à une technique particulière. On peut en donner quelques exemples :

Heuristiques générales :

- quand une situation nouvelle a des analogies avec une situation déjà étudiée, essayer en priorité les méthodes qui ont déjà donné de bons résultats.
- peut-on reposer le problème sous une autre forme.
- peut-on généraliser le problème, (certains problèmes sont plus faciles à traiter sous une forme générale).
- peut-on étudier un cas particulier du problème.

Heuristiques particulières :

- en géométrie :

- si une figure a un axe de symétrie, le tracer.
- si on veut démontrer que deux segments sont égaux, voir s'ils n'appartiennent pas à deux triangles égaux.
- si l'on cherche un lieu, en construire quelques points et voir s'ils ont l'air d'être sur une droite ou sur un cercle. Essayer de démontrer alors qu'ils sont bien sur la ligne conjecturée.

L'emploi des heuristiques ne se limite pas aux mathématiques. On peut en citer quelques uns pour des jeux, par exemple le bridge :

- honneur sur honneur
- cent mille anglais se sont jetés dans la Tamise pour n'avoir pas joué atout assez tôt.
- avant le mort, jouer dans la forte du mort.
- après le mort, jouer dans la faible du mort.
- on ne joue jamais sous son roi.

- - - - - -

La réussite dans un domaine dépend de la qualité des heuristiques que l'on a et de l'habileté avec laquelle on sait choisir en priorité celles qui sont bien adaptées à une situation particulière. Le spécialiste a une grande variété d'heuristiques, qui sont dans certains cas en opposition avec

les heuristiques trop grossières du débutant (cela permet au maitre des échecs de faire des sacrifices incompréhensibles pour le mauvais joueur).

L'utilisation des heuristiques nous parait étrange et peu satisfaisante.

Mais il faut bien voir :

- nous en employons constamment nous même, mais cela parait moins choquant, car la plupart du temps nous ne les formulons pas explicitement.
- d'utiliser des heuristiques ne conduit pas à des résultats faux. Tout au plus cela conduit à ne pas avoir de résultat. Si les heuristiques sont mauvaises, on aura peu de résultats, mais ceux que l'on obtiendra sont aussi valables que si on les avait obtenus à partir des bonnes heuristiques ou à partir d'un algorithme.

L'intuition du mathématicien est essentiellement dûe au fait qu'il a de bonnes heuristiques et qu'il sait choisir celles qui conviennent à la situation. Une étude a été faite sur le joueur d'échec* qui a une démarche intellectuelle très proche de celle du mathématicien. Le grand maitre ne considère pas beaucoup plus d'éventualités que le joueur moyen, seulement il sait choisir les bonnes.

Plusieurs chercheurs ont étudié la possibilité de faire démontrer des conjectures par une machine en utilisant des heuristiques. Dans certains cas, il existait un algorithme, mais ils ne l'ont pas utilisé. On peut citer particulièrement :

- NEWELL, SHAW, SIMON : calcul des propositions. Axiomatique des principia (cf. axiomatique 6).
- GELERNTER : géométrie plane. Le programme avait des performances particulièrement intéressantes dans les problèmes d'inégalités.
- SLAGLE : calcul formel d'intégrales indéfinies.

Les programmes sont beaucoup plus difficiles à écrire, plus lents et moins performants que lorsque l'on utilise un algorithme partiel. Ils ne demandent pas de travail mathématique préliminaire et leurs méthodes sont plus facilement généralisables à d'autres domaines.

Les deux approches ne sont pas toutefois totalement opposées. L'approche heuristique ne donne de solution que pour un certain sous-ensemble de formules, comme avec un algorithme partiel, mais ce sous-ensemble est absolument inconnu et le programmeur n'a pas la moindre idée des théorèmes que son programme pourra démontrer. Certains ont eu des surprises et ont eu des démonstrations qu'ils ne connaissaient pas et qui étaient beaucoup plus élégantes que celles qu'ils connaissaient.

Passons maintenant à l'étude du deuxième type d'activités qui est l'objet de cet essai. Il y a un algorithme qui permet d'énumérer tous les théorèmes : il suffit de combiner les axiomes, puis les théorèmes avec les règles de dérivation pour être sûr d'avoir tous les théorèmes. Ensuite on pourrait adapter un filtre qui mesurerait l'intérêt des théorèmes et on aurait la liste des théorèmes intéressants. Il ne resterait alors qu'un seul problème : définir l'intérêt d'un théorème.

Le schéma est théoriquement parfaitement valable, mais demeure impensable pratiquement, car il demanderait une capacité de mémoire et un temps de calcul qui dépasseraient largement les possibilités des plus puissants calculateurs existants. Il est difficile de chiffrer ceci, mais le nombre des théorèmes d'une axiomatique qui sont sans intérêt est énorme et les théorèmes intéressants sont

---

* aux échecs il existe un algorithme : considérer tous les prolongements possibles, mais une machine 1000 fois plus rapide que la plus rapide calculatrice existant qui aurait commencé à la création de la terre serait loin d'avoir terminé, d'où obligation d'utiliser des heuristiques.

une très petite minorité. Ce phénomène se manifestait très nettement quand on utilisait de mauvaises heuristiques dans un des programmes écrit pour cette étude !

Il ne suffit donc pas d'avoir un filtre qui garde les résultats intéressants, il faut en plus utiliser des heuristiques pour faire d'abord les essais qui ont le plus de chance de donner des résultats intéressants. L'algorithme combinatoire ne peut être utilisé pratiquement, et l'expérience montre qu'on n'est jamais assez sévère dans l'élimination des essais. Même en en gardant une très faible proportion, le nombre de théorèmes augmente très rapidement. On voit les conséquences de mauvaises heuristiques : si elles sont trop sévères, on risque de ne pas faire un essai qui donnerait des résultats importants ; si elles ne sont pas assez sévères, la machine est saturée par un nombre d'essais à faire qui est beaucoup trop grand.

En conclusion :

L'utilisation des heuristiques est fondamentale dans le comportement intelligent ; elle est en général largement inconsciente et baptisée intuition ce qui rend cet emploi moins choquant.

Dans de nombreux cas il n'y a pas d'autre façon d'opérer et même dans les cas où il existe un algorithme, il vaut peut être mieux ne pas l'employer : il peut demander beaucoup de temps et dans beaucoup de cas on se contente très bien d'une bonne solution sans avoir besoin de la meilleure solution.

De toutes les façons une mauvaise utilisation des heuristiques ne conduit pas à des résultats faux, mais à une absence de résultats ce qui est beaucoup moins grave.

Nous n'avons pas abordé le problème capital de la découverte de "bonnes heuristiques". Pratiquement les heuristiques sont soit apprises par un professeur, soit découvertes par hasard et gardées parce qu'elles réussissent, soit trouvées en s'aidant d'heuristique de niveau plus élevé : d'heuristiques qui permettent de trouver de bonnes heuristiques pour un type de problème particulier. On ne fait pas que déplacer le problème, car ces heuristiques d'un niveau élevé sont beaucoup plus générales et sont soit acquises par l'expérience des générations qui nous ont précédé, soit innées. Ce problème n'a, à ma connaissance jamais été abordé et ce n'est pas le sujet de la présente étude. Il convenait toutefois de le signaler au passage et de souligner que les programmes dont on parlera par la suite ont la possibilité d'utiliser des heuristiques qui leur sont imposées, mais ne peuvent pas créer de nouvelles heuristiques. Tout au plus peuvent-ils les adapter si elles comportent des paramètres.

Pour les heuristiques qui ont été employées dans les programmes, on peut donner des arguments de bon sens en leur faveur, mais il est bien évident que l'on ne peut pas donner des raisons qui nécessiteraient leur emploi. Leur seule justification réelle est de donner de bons résultats. Souvent des heuristiques qui, a priori, paraissaient satisfaisantes ont été éliminées, car elles conduisaient à de mauvaises performances.

BIBLIOGRAPHIE

Programmation d'algorithmes ou d'algorithmes partiels

GILMORE

- A program for the production of proofs for theorems derivable within the first order predicate calculus from axioms. Information processing - UNESCO - Paris 1959 p. 265
- A proof method for quantification theory - Its justification and realization. IBM Journal of Research and Development Vol. 4 - n° 1 - p. 28,

D. PRAWITZ, H. PRAWITZ, VOGHERA. - A mechanical proof procedure and its realization on an electronic computer. Journal of the Association for computing machinery. Avril 60 - Vol. 7 - N° 2.

J.A. ROBINSON - Theorem proving on the computer. Journal of the A.C.M. Avril 63 - Vol. 10 - N° 2 - p. 163.

- A machine oriented logic based on the resolution principle. Journal of the A.C.M. - Vol. 12 - N° 1 - Janv. 65 p. 23.

HAO WANG - Towards mechanical mathematics - IBM Journal of Research and Development - Vol. 4 - N° 1 - p. 2.

- Proving theorem by pattern recognition. I Communications of the A.C.M. - Vol. 3 - N° 4 - Avril 60 - p. 220.

- Proving theorems by pattern recognition II The Bell system technical journal - Vol. 40 - N° 1 - Janv. 61 - p. 1.

DAVIS, PUTNAM - A computing procedure for quantification theory. Journal of the A.C.M. Vol. 7 - N° 3 - Juillet 60 - p. 201.

DAVIS, LOGEMANN, LOVELAND - A Machine program for theorem proving. Juin 61 - ASTIA : AD269378.

Généralités sur les heuristiques

POLYA - Mathematical discovery - Vol. I - Wiley and Sons. 1962
- How to solve it. Doubleday Anchor Books - 1957
- L'Heuristique est-elle un sujet d'étude raisonnable. La méthode dans les sciences modernes. Travail et Méthode - Numéro hors série 1958 p. 279.

NEWELL-SIMON - Heuristic problem solving : the next advance in operations research. Operations research - Vol. 6 - N° 1 - Janv. 58 - p. 1.

NEWELL, SHAW, SIMON - Report on a general problem solving program - Information Processing Unesco - Paris - 1959 - p. 256.

- Elements of a theory of human problem solving. Psychological review. Vol. 65 - N° 3 - 1958 - p. 151.

- The processus of creative thinking - Rapport RAND : P. 1320 - Sept. 58.

H. SIMON, P. SIMON - Trial and error search in solving difficult problems : evidence from the game of chess. Behavioral science. Vol. 7 - N° 4 - Oct. 61 - p. 245.

MINSKY - Some methods of artificial intelligence and heuristic programming. Mechanization of throught processes - Vol. 1 - p. 3.

- Heuristic aspects of the artificial intelligence problem. MIT. Group Report 34.55 - Déc. 56.

- Steps toward artificial intelligence. Computers and thought. Edited by Feigenbaum and Feldman Mc Graw Hill - 1963 - p. 406.

GELERNTER, ROCHESTER - Intelligent bchavior in problem solving machines. IBM Journal of Research and development. Vol. 2 - N° 4 - p. 336.

GELERNTER - Realization of a geometry theorem proving machine. Computers and Thought. Edited by Feigenbaum and Feldman - Mc Graw Hill 1963 - p. 134.

GELERNTER, HANSEN, LOVELAND - Empirical exploration of the geometry theorem machine. Computers and thought. Edited by Feigenbaum and Feldman. Mc Graw Hill - 1963 - p. 153.

GELERNTER - Machine generated problem solving graphs. Proceedings of the symposium on mathemetical theory of automata. New-York - Avril 62 - p. 179.

NEWELL - The chess machine : an exemple of dealing with a complex task by adaptation. Proceedings of the Western Joint Computer Conference - Mars 55 - p. 101.

NEWELL, SHAW, SIMON - Empirical exploration of the logic theory machine : a case study in heuristic. Proceedings of the Joint Computer Conference - Fev. 57 - p. 218.

NEWELL, SHAW, SIMON - Chess playing programs and the problem of complexity. Computers and thought. Edited by Feigenbaum and Feldman. Mc Graw Hill - 1963 - p. 39.

SLAGLE - A heuristic program that solves symbolic integration problem in freshman calculus, symbolic automatic integrator. Mai 61 - MIT.

- A heuristic program that solves symbolic integration problems freshman calculus. Computers and thought. Edited by Feigenbaum and Feldman. Mc Graw Hill - 1963 - p. 191.

1C - ASPECT PSYCHOLOGIQUE

Les psychologues se sont assez peu occupés d'étudier la pensée créatrice. Les études les plus intéressantes sont celles de mathématiciens et en particulier Poincaré, Hadamard et Beth.

Ils distinguent quatre étapes :

- **La préparation.** La solution d'un problème ne peut apparaitre que si a eu lieu auparavant un long travail d'acquisition de l'information nécessaire et d'apprentissage.

- **L'incubation.** Durant cette phase, le travail est inconscient. On a l'impression de ne pas avancer vers la solution. Du fait qu'elle est inconsciente, l'introspection ne nous donne absolument aucun renseignement sur ce qui s'y passe. Certains pensent que le travail de l'inconscient est actif, c'est à dire qu'il traite l'information pour faire apparaitre des combinaisons nouvelles. D'autres affirment qu'il est passif, l'inconscient se contentant d'affaiblir la plupart des combinaisons formées au cours du stade précédent ce qui mettra en évidence celle qui reste. Ils s'appuyent sur le fait qu'ils ont remarqué que ce qu'ils trouvent a déjà été noté au cours de la phase précédente, mais sans qu'ils y aient attaché de l'importance. D'autres pensent que le hasard jour un rôle assez important.

De toutes les façons, on en est réduit à des conjectures pour cette phase capitale du fait qu'elle se passe dans l'inconscient. Cela explique d'ailleurs d'une part la difficulté du problème et d'autre part l'impression que l'on a que l'intuition est quelque chose de mystérieux et de non mécanisable alors qu'en réalité il s'agit seulement d'un mode de traitement de l'information qui nous échappe parce que les deux grandes méthodes du psychologue : l'observation d'un sujet et l'introspection n'ont aucune prise sur ce mécanisme.

- **L'inspiration.** Ce n'est pas vraiment une phase, mais une transition extrêmement rapide entre deux phases. Les psychologues américains l'appellent "insight". Tout le monde a fait l'expérience d'une découverte faite en dormant et on peut rappeler la page célèbre de Poincaré où il décrit comment il a été frappé par l'inspiration en montant dans l'omnibus de Coutances. Le sujet a en général une certitude absolue que ce qu'il a trouvé est exact.

- **La vérification et la finition.** Le sentiment de certitude est en général justifié. Il n'empêche qu'il faut quand même vérifier le résultat obtenu. Il faut également exposer les résultats de façon rigoureuse. Au cours de l'inspiration on a les grandes lignes de la démonstration, il reste à établir les intermédiaires. Tous ces calculs demandent un travail conscient qui fait appel à la volonté et à l'attention.

Certains ont essayé pour mieux comprendre le phénomène de poser des problèmes à des sujets en leur demandant d'essayer de les résoudre en pensant à voix haute, c'est à dire en exprimant les idées qui leur venaient. Cette méthode est très intéressante, car il est difficile de faire mieux, mais on en voit les dangers :

- d'une part elle laisse totalement de côté le travail inconscient, puisque le sujet ne peut évidemment énoncer que ses démarches conscientes.

- d'autre part, on n'a accès ainsi qu'à une portion de la pensée consciente, celle qui est verbale. Il est certain qu'il existe une pensée non verbale, qui se passe de langage et utilise des "images" et on ne peut l'exprimer à voix haute. L'existence de la pensée non verbale a été longtemps discutée, probablement parce que certains individus n'ont que la pensée verbale. Mais aussi bien des témoignages (cf. Hadamard), que les études de Piaget ou l'introspection personnelle m'ont convaincu de cette pensée non verbale. Or il est probable que ce mode de pensée joue un rôle très important dans la découverte mathématique.

- Enfin le fait de penser à voix haute perturbe peut être la façon de penser du sujet. Pour penser ainsi, il est obligé de se surveiller constamment, et quand on est vraiment absorbé par un problème complexe, on n'a plus assez de disponibilité pour en même temps s'astreindre à penser à voix haute.

Finalement la pensée à voix haute peut donner des indications sur ce qui se passe dans la résolution des problèmes de petite envergure, où l'on n'a pas besoin de passer par la longue phase d'incubation et pour des individus utilisant la pensée verbale. Elle est bien adaptée pour l'utilisation qu'en ont fait Newell, Shaw, Simon pour étudier la démonstration de théorèmes de logique assez simples. De Groot donne une excellente critique et un bon exemple d'utilisation de la méthode dans le cas des échecs.

A mon avis la méthode la plus efficace est celle où le mathématicien travaille sans se donner la moindre contrainte qui risque de perturber sa pensée, mais en notant tous ses résultats intermédiaires, les conjectures qu'il a posées. Une fois le problème résolu, il reprend son travail et essaye de se rappeler et de comprendre les raisons des diverses étapes. Pour ce qui est inconscient, il ne peut évidemment que conjecturer des causes possibles qui ont permis de passer d'une étape à la suivante. Un exemple d'exposé utilisable pour cela est celui du chemin parcouru par TUTTE et ses amis pour trouver un carré de carrés (dans GARDNER Problèmes et divertissements mathématiques - Tome 2 - Dunod).

Il faut noter que ces études peuvent se faire pour deux buts bien différents.

- On veut comprendre la psychologie du mathématicien. On veut savoir quelles sont réellement les démarches de sa pensée. Une méthode pour résoudre ce problème est de construire un modèle hypothétique et de le tester sur machine. S'il donne des résultats analogues à ceux qu'obtient un mathématicien, le modèle choisi est plausible. Si on a des résultats très différents, le modèle choisi est mauvais. On fait ainsi de la simulation de l'intelligence.

- On veut essayer de réaliser un programme performant, c'est à dire qui donne des résultats si possible meilleurs que ceux qui trouverait un bon mathématicien. On fait alors de l'intelligence artificielle. On ne tient pas à imiter ce que fait l'homme. Bien entendu on s'intéresse à la psychologie, car quand on a la chance d'avoir déjà quelqu'un qui sait faire ce que l'on veut faire faire à la machine, il peut être utile d'essayer de comprendre les mécanismes qu'il utilise pour les reproduire dans le programme s'ils sont efficaces. Mais d'une part on a vu qu'il était très difficile d'avoir une idée sur ces mécanismes par suite de l'importance qu'ont les mécanismes inconscients dans l'invention. D'autre part, il est probable que ces mécanismes ne sont pas très efficaces. L'expérience courante d'un professeur montre bien que l'homme n'est pas tellement doué pour les mathématiques. Il suffit de voir comme des résultats simples ont attendu des siècles avant d'être découverts. Comme les mathématiciens ont été gêné par des considérations extra mathématiques pour leurs découvertes : on connait l'hostilité et la méfiance qui ont suivi la découverte des nombres complexes, les travaux de Cantor. On ne voit pas du reste pourquoi l'homme serait apte aux mathématiques : notre intelligence est le produit de l'évolution, et le développement de la pensée conceptuelle qui permet d'emmagasiner davantage de connaissances et de prédire l'avenir a été à la base de l'importance de l'homme. Accessoirement il a acquis la possibilité de faire des mathématiques, mais la sélection ne joue pas sur les qualités qui font un bon mathématicien : on ne voit pas en quoi elles lui donnent davantage de chances de se reproduire ou de survivre.

C'est pourquoi il est probable que des mécanismes très différents de ceux qu'utilise le mathématicien conduiront à des performances remarquables. Il est même probable que le domaine des mathématiques est le premier où un automate aura des performances supérieures en qualité à celles de l'homme. Il reste quand même la difficulté de découvrir les mécanismes efficaces. Cela ne peut s'acquérir que par tâtonnements : on essaye un mécanisme plausible et on le garde ou on l'élimine selon la valeur des résultats que l'on obtient. L'intelligence artificielle est une science actuellement essentiellement expérimentale : on essaye de découvrir des mécanismes de pensées efficaces par tous les moyens possibles. Evidemment une heuristique largement utilisée est de chercher les causes des actes d'un homme intelligent et de les imiter, mais on ne se borne pas à ceux-ci. Quand on juge des résultats d'un programme, il est capital d'avoir en tête ce que l'auteur a voulu faire : s'il a voulu faire de la simulation de l'intelligence (ex : Newell, Shaw, Simon), il faut voir si les résultats obtenus sont analogues à ceux qu'obtiendrait un homme. S'il a voulu faire de l'intelligence artificielle, il faut voir uniquement la qualité des résultats, le programme n'ayant pas alors la prétention d'être un modèle de comportement humain.

Dans cette étude, on a essentiellement voulu faire de l'intelligence artificielle. Les méthodes utilisées ne sont du reste certainement pas celles employées par l'homme. On a évité en particulier de façon systématique le phénomène si fréquent dans la recherche humaine où l'on refait plusieurs fois la même chose. On a essayé d'utiliser les possibilités qu'apportent la machine par la grande rapidité avec laquelle elle peut manipuler les symboles et par le fait qu'elle ne fait pas d'erreur. Un mathématicien ferait par rapport à notre programme beaucoup moins d'essais, mais choisirait mieux ses essais : les phénomènes de perception, qui permettent au mathématicien de "flairer" l'intérêt d'un essai possible sont parmi ceux qu'on sait le moins bien mécaniser actuellement. Le programme trouve aisément des résultats qui nous paraissent difficiles et ne trouve pas des résultats qui nous semblent plus faciles.

Pour tenter de débroussailler ce problème, il est normal de prendre des théories mathématiques

dont l'énoncé est simple pour limiter la difficulté de l'effort de programmation. Le but n'est pas pour le moment d'avoir des résultats intéressants, mais de voir quelles méthodes donnent des résultats intéressants.

BIBLIOGRAPHIE SOMMAIRE

HADAMARD - Essai sur la psychologie de l'invention dans le domaine mathématique.
Librairie scientifique Albert Blanchard - 1959

BETH, PIAGET - Epistémologie mathématique et psychologique.
P.U.F. 1961

LECLERCQ - La création scientifique - Complément au guide théorique et pratique de la recherche expérimentale.
Gauthier-Villars - 1959.

DE GROOT - Thought and choice in chess.
Mouton - 1965

*

CHAPITRE II

ETUDE LOGIQUE

2A - GENERALITES SUR LES SYSTEMES FORMELS ET NOTATIONS

2A1 - Description d'un système formel

Un système formel (qu'on appelera aussi une axiomatique) comprend :

- un alphabet
- des règles définissant quelles sont les suites de symboles appelées termes
- des règles définissant quels sont les termes appelés théorèmes
- des définitions.

a - **L'alphabet -** Il comprend :

- Un ensemble d'objets que l'on désignera par p, q, r, ... et que l'on appelera variables sans les définir davantage. On supposera cet ensemble illimité : on pourra toujours supposer que l'on peut avoir autant de variables différentes que l'on a besoin (ce nombre étant fini, mais il peut être aussi grand qu'on le veut).

- des opérations ou connectives. A chaque connective est associé un degré qui est un nombre entier positif. Une connective de degré n est dite n - aire ; exemple : connective unaire, binaire ...

b - **Règles définissant les Termes**

On va utiliser une définition récursive :

- une variable est un terme
- si C est une connective de degré n et a_1, a_2 ... a_n sont n termes, alors :

$Ca_1\ a_2 \ldots a_n$ est un terme.

$Ca_1 \ldots a_n$ représentant le concaténation de C suivi des symboles figurant dans a_1 puis des symboles figurant dans a_2 ...

Cette règle est générale et sera utilisée dans tous les systèmes formels que le programme peut traiter.

On a défini ainsi la notation polonaise préfixée. Il ne s'agit pas en réalité d'une limitation très grave, car on peut exprimer le même système formel (c'est à dire qu'il y a correspondance 1-1 entre les théorèmes des différentes formulations) en notation infixée parenthétique, en notation préfixée ou en notation suffixée. L'avantage des deux dernières notations est de supprimer totalement l'emploi des parenthèses. La notation infixée peut être totalement parenthétique, c'est à dire que l'on fait figurer toutes les parenthèses et elle est alors très lourde. La notation classique de l'algèbre permet par des conventions de hiérarchie de connectives de l'alléger en supprimant de nombreuses parenthèses, mais la manipulation des symboles devient très difficile à programmer.

Les notations préfixées et suffixées sont également commodes. La préfixée a été choisie par suite du mode de mémorisation des résultats sous forme d'arborescence : on a alors accès immédiatement à la connective principale qui est en tête.

Donnons à titre d'exemple la même formule d'algèbre dans les diverses notations.

+ × ↑ binaires (↑ : exponentiation)

notation de l'algèbre : x+y↑z+u×(v+w)

totalement parenthétique : ((x+(y↑z))+(u×(v+w)))

préfixée : ++x↑yz×u+vw

suffixée : xyz↑+uvw+×+

Les logiciens utilisent une notation partiellement parenthétique avec hiérarchie d'opérations, les parenthèses étant souvent remplacées par des points.

Ces notations sont souvent peu claires et ambigües, aussi tous les résultats seront exprimés sous forme préfixée, comme ils sont utilisés par la machine. On s'habitue très rapidement à travailler avec la notation polonaise, que du reste certains traités de logique ont déjà utilisée et que la quasi totalité des programmes de démonstration de théorèmes ont employé.

On peut rappeler quelques résultats classiques de la notation polonaise :

- passage de la notation totalement parenthétique à la notation préfixée.

  - au départ on appelle atome une variable
  - si l'on a ox, o étant une opération unaire et x un atome, ox est un atome.
  - si l'on a (xoy), o étant une opération binaire, x et y des atomes, on le remplace par oxy que l'on considère comme un atome.

Exemple - Par commodité on souligne les atomes :

+ : addition
- : soustraction
× : multiplication
/ : division
↑ : exponentiation
θ : moins (unaire)

} binaires

(θx+(y/((z×(x-y))↑(z+x))))
(θx+(y/((z×-xy)↑+zx)))
(θx+(y/(×z-xy↑+zx)))
(θx+(y/↑×z-xy+zx))
(θx+/y↑×z-xy+zx)
+θx/y↑×z-xy+zx

On n'a pas utilisé de connectives ternaires .. qui n'ont pas de représentation en notation infixée et se rencontrent très rarement.

- passage de la notation préfixée à la notation totalement parenthétique. L'algorithme est l'inverse du précédent.

On opère encore de façon récursive :

  - au départ on appelle atome une variable
  - si l'on a ox où o est une opération unaire et x un atome, ox est un atome.
  - si l'on a oxy, où o est une opération binaire et x et y des atomes, on le remplace par (xoy) que l'on considérera comme un atome.

Exemple - Reprenons l'exemple précédent :

$+\theta x/y\uparrow\times z-xy+zx$
$+\theta x/y\uparrow\times z(x-y)(z+x)$
$+\theta x/y\uparrow(z\times(x-y))(z+x)$
$+\theta x/y((z\times(x-y))\uparrow(z+x))$
$+\theta x(y/((z\times(x-y))\uparrow(z+x)))$
$(\theta x+(y/((z\times(x-y))\uparrow(z+x))))$

- Détermination du terme commençant en un certain symbole.

On utilise le théorème fondamental de la notation polonaise, dont on peut trouver la démonstration dans : "The elements of mathematical logic" de Rosenbloom. Il faut d'abord définir le rang d'une suite de symboles. Là encore, on utilise une définition récursive :

Le rang d'une variable est - 1 : $r(p)=-1$
Le rang d'une connective de degré n est (n-1) : $r(C_n)=n-1$

Si A est une suite de symboles et si a est un symbole on a :

$r(Aa)=r(A)+r(a)$.

Théorème : La suite de symboles $a_1 \dots a_p$ est un terme si et seulement si :

1 - $r(a_1 \dots a_p)=-1$.
2 - quelque soit $i<p$ $r(a_1 \dots a_i)\geq o$

Ce théorème est très pratique et est constamment utilisé. Sa programmation est très simple et c'est là que réside un des avantages de la notation polonaise. Pour chercher quel est le terme commençant à un certain symbole on met au départ le rang à 0 et on ajoute au rang de la suite celui des symboles les uns après les autres en s'arrêtant dès que l'on atteint - 1.

Ceci est valable quelque soit le degré des connectives, bien que pratiquement on ne rencontre que des connectives unaires et binaires.

Exemple : ~ : unaire
⊃, ∧, ≡ : binaires
~p ; ~~p ; ⊃~pq ; ~∧≡pqr sont des termes

mais : ⊃pq⊃r n'est pas un terme : condition 2 $r(\supset pq)=-1$
⊃⊃pq n'est pas un terme : condition 1 $r(\supset\supset pq)=0$
≡pq~p n'est pas un terme : condition 2 $r(\equiv pq)=-1$

- recherche du terme commençant au 3ème symbole de :

⊃ ⊃ ~ ⊃ ⊃ p q ~ ⊃ r ⊃ s t u ⊃ v ~ w
0 1 2 1 0 0 1 0 1 0 -1

On met au-dessous des symboles la valeur du rang. On voit que le terme est :

~⊃⊃pq~⊃r⊃st

Remarque : On n'a parlé que de connectives. On aurait pu aussi traiter le cas des relations qui ne pose pas de problème spécial au programme : la seule différence est qu'il y a une relation et une seule par formule car pRq ne peut pas être assimilé à une variable, puisqu'il prend ses valeurs dans un autre espace, alors que pOq, O étant une opération prend ses valeurs dans le même domaine que p et q. Mais le programme n'a pas à s'occuper de ceci qui est automatiquement assuré par les règles de dérivation. On n'a pas envisagé le cas des relations uniquement parce que les axiomatiques traitées n'en avaient pas, mais elles ne poseraient aucun problème.

c - **Règles définissant les théorèmes :**

- un théorème peut être :
  - soit un terme posé au départ comme étant un théorème et que l'on appelle un axiome
  - soit un terme obtenu à partir de théorème déjà dérivé au moyen d'une règle de dérivation (ou production) du type :

Si A et B et C... et L et M sont des théorèmes alors P est un théorème que l'on écrit :

A, B, C, ... L, M $\Rightarrow$ P

Il y a classiquement deux formulations de l'utilisation de l'axiomatique :

- on peut rajouter une règle supplémentaire, dite règle de substitution ; si A est un théorème et b une variable figurant dans A, la suite de symboles B obtenue à partir de A en remplaçant b par un terme t est un théorème.

- on peut utiliser des schémas, chaque schéma d'axiome représentant en réalité une infinité d'axiomes en admettant qu'à la place des variables qui y figurent, on peut mettre n'importe quel terme.

Bien entendu dans les deux cas on fait le même substitution pour toutes les occurrences sans exception de la variable.

On notera une substitution par une succession de groupes du type : a/b, a indiquant la variable pour laquelle on fait la substitution et b étant le terme substitué :

Exemple : Si $CpCqp$ est un théorème, on voit par application de la règle de substitution que :

$CpC\sim qp$	p/p	q/~q
$CpCpp$	p/p	q/p
$CqCpq$	p/q	q/p
$CCpqCrCpq$	p/Cpq	q/r

$CCp\sim\sim qC\equiv r\sim Cqs Cp\sim\sim q$ p/Cp~~q q/≡r~Cqs

sont également des théorèmes.

Mais cela n'entraine pas que $CpCqq$ soit un théorème, car on ne peut pas substituer q pour une seule des occurrences de p.

On démontre aisément (cf. par exemple Church. Introduction to mathematical logic) que ces deux formulations admettent le même ensemble de théorèmes. Il n'y a donc aucune raison de les considérer toutes les deux pour cette étude et nous supposerons par la suite que nous avons des schémas d'axiomes.

Voyons maintenant l'utilisation des productions. Dans la production A, B, C ... L, M $\Rightarrow$ P les termes A, B ... M sont appelés les antécédents. S'il y en a n, la production sera dite à n antécédents. Pratiquement on rencontre généralement des productions à 1 ou 2 antécédents. Le programme n'accepte pas des productions à plus de 2 antécédents. P est le résultat de la production.

Exemple : Supposons que l'on ait les théorèmes :

$CpCqp$

$CCpqCCqrCpr$

et la production à 2 antécédents $p, Cpq \Rightarrow q$.

on voit qu'en faisant dans la production la substitution :

$p/CpCqp$ $q/CCCqprCpr$ elle devient :

$CpCqp, CCpCqpCCCqprCpr \Rightarrow CCCqprCpr$

Mais $CpCqp$ est un théorème

ↃↃpↃpqↃↃↃqprↃpr est un théorème puisqu'on l'obtient à partir de ↃↃpqↃↃqrↃpr par la substitution : p/p q/Ↄqp r/r

Donc ↃↃↃqprↃpr est un théorème.

Remarque On utilisera constamment une forme canonique d'énonciation des théorèmes où l'on appelle p la première variable rencontrée, puis q la première qui suit et différente de p ... Cela se fait par une simple permutation du nom des variables et le programme a un sous-programme très simple qui fait systématiquement ce travail.

Exemple : ↃↃↃqprↃpr devient ↃↃↃpqrↃqr par p/q q/p r/r.

Dans la suite nous ferons automatiquement cette transformation sans l'indiquer.

On peut donner des exemples de productions : on vient de voir un exemple de production à deux antécédents : p, pq⇒q. Montrons en une à un antécédent.

Soit : ↃↃpqr⇒Ↄqr et supposons que ↃↃpↃqrↃↃpqↃpr soit un théorème. On voit que par la substitution :

p/p q/Ↄqr r/ↃↃpqↃpr la production devient :

ↃↃpↃqrↃↃpqↃpr⇒ↃↃqrↃↃpqↃpr

Mais l'antécédent est un théorème, donc ↃↃqrↃↃpqↃpr (ou ↃↃpqↃↃrpↃrq) est un théorème.

d - **Les définitions**

Il est très fréquent, pour éviter d'avoir des suites de symboles trop longues, d'utiliser un procédé qui permet de remplacer une suite de symboles qui apparait souvent par une autre suite plus courte.

Dans une définition, on pose la suite $a_1\ a_2 \ldots a_p$ équivalent à la suite $b_1 \ldots b_q$ la première suite comportant au moins un symbole nouveau. On l'écrira : $a_1\ a_2 \ldots a_p = b_1 \ldots b_q$

Les a_i et b_i comportent des connectives et des variables. On peut à une variable substituer un terme, à condition qu'on le fasse pour toutes les autres occurrences de cette variable dans les **deux** suites.

Ce procédé parait assez proche des productions. Il en diffère pourtant par de nombreux points et la définition précédente n'est pas équivalente à la production

$a_1 \ldots a_p \Rightarrow b_1 \ldots b_q$.

- l'utilisation de la définition est réversible.
  A la suite $a_1 \ldots a_p$ on peut substituer $b_1 \ldots b_q$ et à $b_1 \ldots b_q$ on peut substituer $a_1 \ldots a_p$.
- On ne demande pas que $a_1 \ldots a_p$ soit un théorème, ce peut être seulement un terme qui figure dans un théorème.

Si l'on a $Aa_1 \ldots a_pB$ est un théorème on en déduit
$Ab_1 \ldots b_qB$ est un théorème A et B étant deux suites de symboles quelconques (pouvant être éventuellement vides). La production ne nous permet de faire ceci que si A et B se réduisent toutes deux à la suite vide.

Exemples de définition :

$\wedge pq = \sim\supset p\sim q$: $\supset\sim\supset p\sim qp$ est le même théorème que $\supset\wedge pqp$

$\equiv pq = \wedge\supset pq\supset qp$: $\supset\wedge\supset pq\supset qp\supset pq$ est le même théorème que $\supset\equiv pq\supset pq$

$\vee pq = \supset\sim pq$: $\supset\Sigma\sim ppp$ est le même théorème que $\supset\vee ppp$

$\sim p = |pp$: $|p|pp$ est le même théorème que $|p\sim p$

L'emploi des définitions n'apporte théoriquement pas grand chose, mais est pratiquement très important, en réduisant la taille des formules et en faisant apparaitre les théorèmes sous un jour plus intuitif.

2A2 - Présentation, Représentation, Interprétation

On peut distinguer trois niveaux dans la description d'un système formel.

- Le premier niveau est celui de la présentation. C'est celui qui vient d'être adopté. On ne se préoccupe absolument pas du sens que l'on peut donner aux diverses connectives, on s'intéresse uniquement aux règles qui les régissent. Le choix des symboles pour représenter les opérations du système est conventionnel. On n'a également aucune correspondance avec le monde extérieur. Ce niveau est purement syntaxique et c'est celui qui sera adopté durant toute cette étude.

- Le deuxième niveau est celui de la représentation. On donne un sens bien déterminé aux opérations et aux variables du système : On peut dire par exemple : les variables sont des propositions, la connective ~ est la négation, la connective $\supset$ est l'implication.

Pour des commodités d'exposition, il nous arrivera de désigner en utilisant leur nom usuel les connectives utilisées. Mais le programme travaille au niveau précédent et pour lui le symbole $\supset$ n'est pas l'implication, mais un symbole qui obéit à certaines règles et que l'on pourrait aussi bien appeler la multiplication.

Ce niveau est beaucoup moins important que les deux autres. On a l'habitude de l'utiliser et on doit l'employer pour exprimer certaines notions comme la non contradiction ou la réfutation qui supposent que l'on a distingué dans la langue une certaine connective unaire que l'on appelle négation. Ces notions très classiques méritent peut être qu'on leur attache moins d'importance que celle qui leur est habituellement donnée.

- Le troisième niveau est celui de l'interprétation. On met en correspondance les termes du système avec des énoncés dont la vérité ou la fausseté est déterminée indépendamment du système. Il faut naturellement, pour que cela ait un intérêt, s'arranger pour qu'aux théorèmes correspondent des énoncés vrais. Cette correspondance est appelée une interprétation du système formel. Le modèle est l'ensemble des énoncés. Un problème important est de savoir s'il existe ou non des modèles du même système formel qui ne soient pas isomorphes, les théories les plus riches étant celles pour lesquelles il existe plusieurs modèles non isomorphes. Un autre problème est de déterminer si toutes les propositions vraies sont dérivables. Tarski a démontré qu'il existait des systèmes formels pour lesquels il n'existait pas de modèle où toutes les propositions vraies soient dérivables.

Ce niveau fait appel à la sémantique. Dans la présente étude, nous n'y avons absolument pas fait appel. La logique des propositions devient du reste trop facile quand on utilise les valeurs de vérité.

C'est pour cela que les études de démonstrations de théorèmes sur machine à l'aide d'heuristiques ont rarement fait appel à la sémantique. On peut toutefois citer l'étude remarquable de Gelernter pour la démonstration de théorèmes de la géométrie plane qui utilise un modèle.

En conclusion nous nous bornerons au niveau de la présentation, en utilisant dans l'exposition, par abus de langage, des notions qui relèvent de la représentation. Nous ne ferons absolument pas appel à la sémantique.

2A3 - Métalangues

Quand nous avons défini les productions, nous avons fait apparaitre des signes nouveaux : ", " qui sépare les antécédents et "⇒" qui sépare le dernier antécédent du résultat. On a vu que A, B ⇒ C se lit :

Si A est un théorème et si B est un théorème alors C est un théorème. Donc ", " est une opération " et " et "⇒ " une implication. Mais il faut faire attention de ne pas les confondre avec des opérations qui portent le même nom et que l'on désigne par "∧" et "⊃".

Ces opérations ", " et "⇒ " sont en réalité les opérations d'un nouveau système formel que i'on appelle métalangue (ou Mlangue) et dont les théorèmes, appelés métathéorèmes (ou Mthéorèmes) pour les distinguer agissent sur les suites de symboles du système de départ.

On a ainsi deux niveaux de langage, un niveau nous donnant les théorèmes que l'on cherche, le niveau au-dessus créant des métathéorèmes, qui peuvent être en particulier de nouvelles productions qui vont faciliter les démonstrations des théorèmes ou énoncer que certains ensembles de termes sont des théorèmes :

- exemple : "les formules $\supset\sim^{2n}pp$ sont des théorèmes de tel système formel". C'est un métathéorème pour certains systèmes formels.

- autre exemple : Supposons qu'on ait un système formel qui ait les productions :

$\supset p\supset qr \Rightarrow \supset\supset pq\supset pr$
$\supset\supset pqr \Rightarrow \supset qr$

Supposons qu'on ait un théorème de la forme $\supset P\supset QR$; P, Q et R étant des termes. D'après la première production $\supset\supset PQ\supset PR$ est un théorème et d'après la deuxième production $\supset Q\supset PR$ est un théorème. On voit donc que si $\supset P\supset QR$ est un théorème alors $\supset Q\supset PR$ est un théorème :

On vient de démontrer un métathéorème : $\supset p\supset qr \Rightarrow \supset q\supset pr$ qui est une nouvelle production.

Pour le démontrer nous avons utilisé un langage non formalisé où l'on fait appel à des règles supposées admises par tous que l'on n'a pas pris la peine de préciser.

On a en réalité appliqué la règle suivante :

Si $a \Rightarrow b$ est une production et si $b \Rightarrow c$ est une production, alors $a \Rightarrow c$ est une production.

On voit apparaitre le "si...alors" et le "et" catactéristiques d'opérations logiques. Mais ces opérations appartiennent à la métalangue de la métalangue et si on veut les formaliser il faut créer des symboles nouveaux qui seront appelés MM connectives pour bien rappeler qu'ils appartiennent à la métalangue de la métalangue (ou MMlangue).

on utilisera " ; " pour le " et " de la MMlangue
"$\longrightarrow$" pour l'implication de la MMlangue.

et l'on écrira la règle précédente
$a \Rightarrow b$; $b \Rightarrow c \longrightarrow a \Rightarrow c$

qui exprime tout simplement la transitivité de l'implication de la métalangue (ou Mlangue).

Mais on peut recommencer la même opération : supposons que nous ayons la production p, ⊃pq⇒q. On voit que si ⊃pq est un théorème, on peut en déduire le Mthéorème : p⇒q, on a donc établi le méta-métathéorème (ou MMthéorème) : ⊃pq⟶p⇒q.

De même supposons que l'on ait le Mthéorème p⇒⊃qr (c'est à dire que si p est un théorème, alors ⊃qr est un théorème); on sait que si ⊃qr et q sont des théorèmes, alors r est un théorème. Donc si p⇒⊃qr est un Mthéorème, si p et q sont des théorèmes, alors r est un théorème. On a établi le MMthéorème :

$$p\Rightarrow\supset qr\longrightarrow p,q\Rightarrow r.$$

On voit la difficulté du raisonnement non formalisé.
Il y a deux "si" dans la phrase précédente, le premier appartient à la MMlangue et le second à la Mlangue.

Comme on l'a fait précédemment, on peut essayer de formaliser les raisonnements qui nous ont conduit à ces MMThéorèmes. Il faut considérer la MMMlangue et des MMMconnectives. Soit "⇝" l'implication de la MMMlangue.

on a dans le premier cas utilisé la règle :

$$a,b\Rightarrow c\rightsquigarrow b\longrightarrow a\Rightarrow c$$

et dans le deuxième cas :

$$a,b\Rightarrow c\rightsquigarrow p\Rightarrow b\longrightarrow p,a\Rightarrow c$$

p étant une variable ne figurant pas dans a, b ou c.

L'utilisation des productions de la MMMlangue est un outil extrêmement puissant, car cela permet d'engendrer des MMThéorèmes qui à leur tour engendreront des Mthéorèmes (ce seront en général des productions) qui fournissent finalement les théorèmes voulus.

Leur intérêt réside dans leur généralité. On voit dans les deux règles précédentes que n'y figure aucune connective, mais des Mconnectives, des MMconnectives ou des MMMconnectives : **elles sont donc totalement indépendantes d'un système formel particulier, mais engendreront des MMthéorèmes qui eux seront liés à un système formel particulier.**

On a programmé uniquement des productions de la MMlangue ou de la MMMlangue qui ne comportaient pas de connectives de la langue pour être assuré justement de cette généralité.

On voit que la deuxième règle engendrera :

Si p,⊃pq⇒q est une production, le MMthéorème : p⇒⊃qr⟶p,q⇒r.

Si p,|p|qr⇒r est une production, le MMthéorème : p⇒|q|rs⟶p,q⇒s.

Nous donnons ci-après un tableau où nous rappelons les diverses connectives utilisées pour les divers niveaux linguistiques.

	langue	Mlangue	MMlangue	MMMlangue
implication	⊃	⇒	⟶	⇝
et	∧	,	;	□
identité	=	⇔	⟷	

Si I_n est l'implication de la M_n langue
E_n est l'identité
A_n est le "et"

E_n est défini par :

$$pE_nq = (pI_nq)A_n(qI_np)$$

Pour la langue on considère d'autres connectives qui correspondent dans la représentation habituelle des systèmes formels de la logique :

unaire : $\sim$: négation
$\Diamond$: possibilité
N : Nécessité
binaire : $\vee$: ou
| : Sheffer's stroke (ou incompatibilité)

On peut considérer une hiérarchie infinie de métalangues. Nous nous sommes arrêtés à la MMMlangue pour deux raisons :

- on n'a pas l'esprit assez puissant pour bien voir ce qui se passe à des niveaux métalinguistiques très élevés, et on à peine à comprendre ce que l'on fait.

- il semble inutile d'utiliser un niveau supérieur à partir du moment où les M^n Théorèmes que l'on a ne comportent plus d'opérations de la langue : ils ont ainsi un grand caractère de généralité et on peut les programmer. Dans le cas présent on a programmé des MMthéorèmes (comme $a \Rightarrow b; b \Rightarrow c \longrightarrow a \Rightarrow c$) et des MMMthéorèmes (comme $a, b \Rightarrow c \rightsquigarrow b \longrightarrow a \Rightarrow c$).

Les MMMthéorèmes sont tous imposés au programme.

Les MMthéorèmes sont soit trouvés par application des MMMthéorèmes, soit lus (ce qui permettrait théoriquement de les changer; effectivement au cours des essais on en a supprimé un qui semblait ne jamais donner de résultats intéressants et on en a ajouté pour certains essais).

Les Mthéorèmes sont **tous** trouvés par le programme, ainsi naturellement que les théorèmes.

On donnera en 2D la liste des MMMthéorèmes et des MMthéorèmes utilisés par le programme.

Remarque 1 - Une production de la M^nlangue est un théorème de la M^{n+1}langue, mais un théorème de la M^{n+1}langue a n'est pas forcément une production de la M^nlangue.

Si l'on considère le système formel qui a les connective U unaire et B binaire, pour axiome Bpp et pour production $Bpp \Rightarrow BUpUp$.

L'énoncé : "tout thèorème de ce système formel est de la forme $BU^npU^np, n=0,1..$" est un Mthéorème qui n'est pas une production.

Le programme n'a la possibilité de considérer que des Mthéorèmes qui sont des productions ce qui en limite la puissance. Il est difficile de créer d'autres sortes de Mthéorèmes, car il y en a une très grande variété et cela exigerait d'avoir une MMlangue beaucoup plus puissante que celle que l'on a envisagé. La formalisation que l'on en a faite ne permet pas en particulier de formaliser l'énoncé donné en exemple.

Remarque 2 - L'utilisation de la notation polonaise ne se fait qu'au niveau de la langue. Pour les niveaux supérieurs les formules sont écrites en notation infixée sans parenthèses. On utilise la hiéarchie évidente : une M^nconnective a un degré dans la hiérarchie plus élevée qu'une M^{n-1}connective.

Cela n'amène pas d'ambiguité, car les formes des Mthéorèmes sont très stéréotypées. C'est ainsi qu'on considère seulement les Mthéorèmes des types : a⇒b ou a,b⇒c c'est à dire les productions à 1 ou 2 antécédents.

De même les MMthéorèmes que le programme peut découvrir n'ont que deux types possibles prévus :

$$a \longrightarrow b \Rightarrow c \text{ ou } a \Rightarrow b \longrightarrow c, d \Rightarrow e.$$

Il est bien évident que a,b⇒c se comprend (a,b)⇒c et non a,(b⇒c) qui n'aurait pas de sens.

Pratiquement, le programme n'utilise pas les Mconnectives et les MMconnectives du fait qu'il sait toujours de quel type est le Mthéorème ou le MMthéorème qu'il utilise. Cela ne poserait pas de problème de les mettre explicitement, mais prendrait de la place sans rien apporter.

S'il avait des possibilités encore plus évoluées d'opérer à ces niveaux, il serait alors indispensable de les mettre explicitement et d'utiliser soit des parenthèses, soit la notation polonaise.

2A4 - Théorèmes de limitation et intelligence artificielle

Il est intéressant de voir un peu plus près les conséquences des théorèmes de limitation. En effet les logiciens, qui les ont démontrés, en ont presque tous tiré des conclusions inexactes sur les possibilités de l'intelligence artificielle.

Les théorèmes de limitation sont de trois ordres.

a - Théorèmes liés à la présentation

Ils ne font appel qu'à la dérivabilité. De ce type est le théorème de Church ou les résultats d'indécidabilité obtenus dans la théorie de la machine de Turing. Ils expriment que dans certaines axiomatiques, il n'existe pas de procédé systématique qui permette en une suite d'opérations finie de décider si une suite de symboles est ou n'est pas dérivable à partir des axiomes en utilisant les productions. Le problème est alors dit indécidable.

La raison fondamentale est que l'ensemble des théorèmes est toujours récursivement dénombrable, c'est à dire que l'on peut énumérer tous les théorèmes, mais n'est pas toujours récursif ; dans ce cas cela signifie que l'ensemble des énoncés qui ne sont pas des théorèmes n'est pas récursivement dénombrable : on ne peut pas énumérer tous les énoncés qui ne sont pas des théorèmes. Quand on a un énoncé, si c'est un théorème, on l'obtiendra surement en un temps fini par l'énumération des théorèmes, mais si ce n'est pas un théorème, on ne l'obtiendra jamais et on ne saura pas si c'est dû au fait que l'on n'a pas encore énuméré assez de théorèmes ou si c'est parce que ce n'est pas un théorème.

Donnons un exemple pour préciser ceci. Supposons que l'on considère comme théorèmes toutes les suites de chiffres qui sont comprises entre deux occurrences consécutives de 0 dans la suite des décimales de π. On peut facilement énumérer les théorèmes en calculant toutes les décimales et chaque fois que l'on a un zéro, commence l'énoncé d'un nouveau théorème. Par contre, on ne voit pas comment être sûr si une suite de chiffres donnée va se trouver comprise entre deux zéros ou non. Si on a de la chance, on va peut être la trouver rapidement et on saura que c'est un théorème. Sinon après avoir calculé un milliard de décimales, on ne sait toujours pas si on l'aura dans quelques milliards de décimales ou si on ne l'aura jamais.

Mais le fait que le problème soit indécidable ne nous rend pas totalement désarmés. Il se peut

très bien que l'on sache le résoudre dans des cas particuliers. Dans l'exemple précédent on sait que la suite "9805" n'est pas un théorème puisqu'elle contient un zéro, et ne peut donc se trouver entre deux zéros consécutifs. On vient de démontrer un métathéorème de cette théorie : un théorème ne contient pas le symbole 0. Justement, les logiciens essayent, dans les théories indécidables, de définir des sous-ensembles d'énoncés pour lesquels ils savent résoudre le problème de la décision.

Dans l'exemple précédent on verrait que l'on a :

- tout énoncé composé d'un seul chiffre différent de 0 est un théorème (parce qu'il y a dans la suite des décimales au moins une fois 010, 020, --- 090).

- tout énoncé comprenant au moins un 0 n'est pas un théorème.

ce qui fait deux classes de formules pour lesquelles le problème de la décision a été résolu.

Les méthodes heuristiques définissent aussi des classes d'énoncés pour lesquels elles résoudront le problème. Seulement alors que dans les exemples précédents, ces classes étaient parfaitement définies, la classe des énoncés dont on pourra montrer qu'ils sont dérivables ou qu'ils ne sont pas dérivables n'apparait pas clairement à première vue. Mais dans le cas où le système est indécidable, on est sûr qu'elle est incluse strictement dans l'ensemble des énoncés ; ceci est valable pour le mathématicien aussi bien que pour la machine. Le mathématicien n'a à sa disposition qu'un ensemble fini de méthodes heuristiques qui vont guider ses essais et tout ce qu'il peut faire est forcément inclus dans une utilisation combinatoire de ces heuristiques. S'il avait la possibilité de résoudre le problème de décision, c'est que ce problème ne serait pas indécidable, puisqu'un procédé combinatoire permettrait de la résoudre.

b - Théorèmes liés à la représentation

On fait jouer un rôle particulier à une connective unaire que l'on appelle négation. On introduit la notion de réfutation : une suite de symboles $a_1 \ldots a_n$ est réfutable si $\sim a_1 \ldots a_n$ est dérivable.

Il faut bien voir que cette notion est différente de celle de non dérivabilité : quand on dit que $a_1 \ldots a_n$ n'est pas dérivable, la négation introduite par les mots "ne...pas" est celle de la Mlangue. Quand on dit que $\sim a_1 \ldots a_n$ est dérivable, on a la négation de la Mlangue. Bien entendu, les systèmes formels ayant un but pratique, il faut une certaine correspondance entre les deux négations, et on s'arrange pour que si $a_1 \ldots a_n$ est dérivable, $\sim a_1 \ldots a_n$ ne soit pas dérivable (c'est une définition de la cohérence, propriété que l'on exige des systèmes formels pour les considérer comme utiles.)

Un théorème de limitation lié à la représentation est le théorème de Gödel : il existe dans certains systèmes formels des suites de symboles telles qu'elles ne soient pas dérivables et que leur négation ne soit pas dérivable.

A priori il y a quatre solutions possibles pour une suite de symboles :

- elle est dérivable et sa négation n'est pas dérivable.
- elle n'est pas dérivable et sa négation est dérivable.
- elle n'est pas dérivable et sa négation n'est pas dérivable.
- elle est dérivable et sa négation est dérivable.

On empêche le quatrième cas de se produire en introduisant une condition de cohérence. Si cela se produit ; le système formel ne sera pas étudié.

L'importance du résultat de Gödel est de montrer que dans des systèmes utilisés dans la pratique, il existe des formules qui appartiennent au troisième cas. Cela n'a au fond rien d'étonnant. On ne voit pas pour quelle raison quelle que soit la suite de symboles Pobéissant aux règles de la

syntaxe on devrait pouvoir dériver soit P soit P précédé d'une certaine connective unaire. On n'arrive pas dans les systèmes "puissants", donc complexes, à avoir toutes les formules dans les deux pre-premières classes.

La démonstration de Gödel est constructive : elle donne effectivement une suite de symboles qui appartient à cette troisième classe (elle est basée sur une formalisation du célèbre paradoxe du menteur).

c - Théorèmes liés à l'interprétation

On a choisi un modèle pour notre système formel, c'est à dire que l'on a établi une correspondance entre les symboles et des éléments du monde extérieur. En particulier on a souvent une loi qui permet de décider si l'on attribue à une suite de symboles la valeur "vrai" ou "faux".

Comme dans le cas précédent, quatre cas sont possibles pour une suite de symboles :

- elle est dérivable et sa valeur est vrai.
- elle n'est pas dérivable et sa valeur est faux.
- elle n'est pas dérivable et sa valeur est vrai.
- elle est dérivable et sa valeur est faux.

Bien entendu, si le quatrième cas se produit, c'est que l'on a mal choisi notre interprétation et il faut en changer : le bon sens n'accepte pas d'avoir des théorèmes qui soient faux !

Par contre, le théorème de Tarski montre que pour certaines axiomatiques assez puissantes, il n'est pas possible de trouver d'interprétation telle que la troisième classe soit vide. Il existe des suites de symboles qui ne sont pas des théorèmes et auxquelles pourtant notre interprétation donne la valeur vrai.

On ne voit pas pour quelle raison cette classe n'existerait pas. Tarski donne du reste un excellent exemple qui rend plus concret son théorème. Dans un système formel qui a des interprétations arithmétiques, il considère la formule : $(y)(Ex)x+x=y$.

(y) est le quantificateur universel. (Ex) celui d'existence.

Il montre que le système formel a deux interprétations arithmétiques, l'une où les variables représentent des entiers et l'autre des rationnels.

Il est bien évident que la formule va être fausse dans le premier cas, puisque, pour un entier impair, il n'existe pas d'entier qui en soit la moitié, alors qu'elle sera vraie dans le deuxième cas.

Cet énoncé et sa négation ne sont pas dérivable puisque l'on a supposé que les deux interprétations étaient correctes donc que pour eux la quatrième classe est vide. On voit que pour ces deux interprétations, la troisième classe ne sera pas vide.

- pour la première $\sim(y)(Ex)x+x=y$ sera dans la troisième classe
- pour la seconde $(y)(Ex)x+x=y$ sera dans la troisième classe.

Cet exemple n'a pas la prétention d'être une démonstration, il vise seulement à montrer par des considérations purement intuitives que les phénomènes de limitation n'ont rien d'extraordinaire.

d - Conclusion

En définitive on voit que ces théorèmes de limitation (on n'a parlé ici que des plus importants,

mais les autres n'apportent rien de plus pour ce qui nous concerne) ne donnent aucune limitation qui ne soit valable que pour les machines. Le mathématicien y est également soumis.

Les théorèmes du premier type sont liés au fait qu'il n'existe pas de procédé systématique qui permette de montrer à coup sûr en un nombre fini d'étapes si un résultat donné est ou non dérivable. Il faudrait prouver que le mathématicien puisse employer autre chose qu'un procédé systématique pour avoir le droit de conclure à une supériorité de celui-ci. Un procédé systématique utilisant des heuristiques semble donner des résultats proches de ce que fournit l'intuition du mathématicien.

Les théorèmes du deuxième type montrent qu'il y a des résultats qui ne sont ni dérivables, ni réfutables : il n'existe pas de dérivations qui lient les axiomes à l'énoncé ou à sa négation, et le mathématicien ne pourra pas évidemment les trouver mieux que la calculatrice, puisqu'elles n'existent pas.

Les théorèmes du troisième type montrent qu'il y a des propositions vraies qui ne sont pas dérivables. Là encore, ni le mathématicien, ni la calculatrice ne peuvent trouver une dérivation qui n'existe pas.

Il est manifeste qu'il n'y a aucune infériorité de la calculatrice pour les deux derniers résultats Et pourtant le théorème de Gödel a été maintes fois cité pour condamner l'intelligence artificielle.

Par contre il peut y avoir doute pour le premier type de résultats, comme le théorème de Church ou des théorèmes d'indécidabilité de la machine de Turing ou des grammaires formelles. Mais il ne peut y avoir supériorité de l'homme que si ce dernier est capable d'obtenir des résultats qui ne sont pas calculables par une machine de Turing. Si l'on n'a pas démontré que c'est faux, on n'a pas non trouvé d'exemple où c'était effectivement réalisé. Bien au contraire, l'idée de calculabilité par la machine de Turing semble satisfaisante pour notre intuition et la convergence de cette définition avec celle des fonctions récrusives montre qu'il s'agit de quelque chose de profond.

Si les logiciens ont très souvent condamné l'intelligence artificielle à l'aide de ces résultats, ceux qui étudient l'intelligence artificielle sont d'accord pour refuser cette condamnation. On peut lire en particulier l'article de CANNONITO : "The Gödel incompleteness theorem and intelligent machines" Proceedings of the Spring Joint Computer Conference - 1962 - p. 71.

Un des meilleurs exposés sur les théorèmes de limitation est celui de LADRIERE : Les limitations internes des formalismes. Gauthier Villars.

2B - ALGORITHME DE DERIVATION

Si l'on veut appliquer la production b⇒c (si b est un théorème, alors c est un théorème), il faut par des substitutions amener b à coïncider avec un théorème. Par exemple on peut l'amener à être le théorème a ou un théorème issu de a par des substitutions.

Exemple : a est ⊃p⊃qp

b⇒c est : ⊃⊃pqr⇒⊃qr

si l'on fait dans a la substitution : p/⊃pq;q/q et dans b⇒c la substitution p/p;q/q;r/⊃q⊃pq on voit que a devient : ⊃⊃pq⊃q⊃pq et b⇒c : ⊃⊃pq⊃q⊃pq⇒⊃q⊃q⊃pq.

Donc ⊃q⊃q⊃pq (ou ⊃p⊃p⊃qp) est un théorème.

Mais on aurait pu faire les substitutions :

dans a : p/⊃pq;q/r

dans $b \Rightarrow c$: p/p ; q/q ; r/$\supset r \supset pq$

et l'on verrait que $\supset q \supset r \supset pq$ (ou $\supset p \supset q \supset rq$) est aussi un théorème. Ce théorème est plus général que le premier, car on peut en déduire le premier par la substitution : p/p ; q/q ; r/q.

Il est intéressant de trouver un algorithme qui obtienne le résultat le plus général possible à partir du théorème a et de la production $b \Rightarrow c$ quand on essaye, par des substitutions dans b, à l'amener à être identique à un théorème se déduisant de a par des substitutions. Il est sans intérêt de considérer d'autres substitutions qui ne donneraient que des résultats plus faibles. Nous ne ferons que les substitutions strictement indispensables pour amener a et b à coïncider et nous obtiendrons ainsi une sorte de p.p.c.m, tous les résultats possibles s'en déduisant par substitutions.

Nous supposons au départ qu'il n'y a aucune variable commune à "a" et à "$b \Rightarrow c$" ce qui est possible puisque nous avons supposé que nous aurions toujours autant de variables que nous le désirerions. Il faut les supposer différentes, car l'hypothèse contraire revient à faire une substitution qui n'est pas indispensable.

Supposons que les p premiers symboles (p peut être nul) de a et de b coïncident :

$a = a_1 \ldots\ldots a_p \; a_{p+1} \ldots\ldots a_l$

$b = a_1 \ldots\ldots a_p \; b_{p+1} \ldots\ldots b_m$

$c = c_1 \ldots\ldots c_n$

On va comparer a_{p+1} et b_{p+1}. 3 cas sont possibles :

1 - a_{p+1} et b_{p+1} sont des connectives. Si ce sont les mêmes, les p+1 premiers symboles coïncident et on examine le symbole suivant. Si elles sont différentes, aucune substitution ne peut amener deux connectives à être identiques : elle ne font que remplacer des variables par des termes et ne changent pas les connectives. Il y a alors échec : il n'existe aucune substitution qui puisse amener des résultats obtenus à partir de a et b à coïncider .

2 - a_{p+1} et b_{p+1} sont des variables. Si ce sont les mêmes (ce n'est pas possible au départ, mais le jeu des substitutions peut faire apparaitre dans a des variables de b et inversement), on passe au symbole suivant. Si elles sont différentes on substitue à la variable b_{p+1} la variable a_{p+1}. Cette substitution est évidemment toujours possible. La substitution se fait dans $a_{p+2} \ldots\ldots a_l$; $b_{p+2} \ldots\ldots b_m$; $c_1 \ldots\ldots c_n$. Il est inutile de le faire dans $a_1 \ldots\ldots a_p$ qui coïncide déjà pour a et b et qui est sans intérêt : le résultat intéressant est c qui donnera finalement le théorème désiré.

3 - L'un est une variable et l'autre une connective. Le raisonnement étant le même dans les deux cas, supposons que a_{p+1} soit une variable et b_{p+1} une connective. Soit $t = b_{p+1} \; b_{p+2} \ldots\ldots b_{p+k}$ le terme qui commence en b_{p+1} (on a vu en 2A un algorithme qui le détermine). Pour amener a et b à coïncider il faut substituer t à toutes les occurrences de la variable a_{p+1} dans :

$a_{p+2} \ldots\ldots a_l$

$b_{p+k+1} \ldots b_m$

$c_1 \ldots\ldots c_n$

Il est inutile de le faire dans $a_1 \ldots\ldots a_p$, qui, coïncidant déjà, seront encore identiques après substitution.

Il faudrait également faire la substitution dans t s'il n'y trouve des occurrences de a_{p+1}. Mais il y a alors échec, car il est impossible de faire coïncider par des substitutions une variable et un terme qui contient cette variable : aucune substitution ne peut amener p à coïncider avec $\supset pq$. Si on modifie le premier, on modifie le second aussi et il reste encore une différence.

Les autres substitutions sont toujours possibles : rien n'empêche de substituer un terme à une variable.

S'il n'y a pas eu échec, on reprendra le même procédé en partant de a_{p+2} et b_{p+2} dans les deux premiers cas, de a_{p+2} et b_{p+k+1} si l'on est dans le 3ème cas et si a_{p+1} est une variable et de a_{p+j+1} et b_{p+2} si b_{p+1} est une variable et $a_{p+1}.....a_{p+j}$ le terme commençant en a_{p+1}.

L'algorithme s'achève quand les suites a et b sont épuisées. Elles le sont simultanément puisque le procédé précédent assure que les symboles étudiés simultanément ont toujours le même rang au sens de la notation polonaise : si on a 2 connectives, elles doivent être identiques et ont alors le même degré. Si on a 2 variables ou une variable et un terme, le rang variera dans les deux cas de -1 qui est le degré des variables et des termes.

Le résultat sera alors la suite c modifiée par toutes les substitutions. Ce sera le résultat le plus fort possible puisque l'on n'aura fait que les substitutions indispensables.

Nous représenterons l'application de cet algorithme par :

a+b⇒c⊣⊢d qui correspond au MMthèorème:
a ; a⇒b⟶b

Pratiquement, on peut, pour représenter les étapes de l'algorithme, mettre sur une ligne la partie de a qu'il reste à traiter, sur une autre ligne la partie correspondante b et c sur une troisième ligne. Pour distinguer les variables, nous mettrons des minuscules pour celles de a et des majuscules pour celles de b⇒c.

Exemple : ⊃p⊃qp+⊃⊃pq⊃⊃r⊃st⊃rs⇒⊃⊃r⊃st⊃q⊃rt⊣⊢?

1 - ⊃p⊃qp
⊃⊃PQ⊃⊃R⊃ST⊃RS
⊃⊃R⊃ST⊃Q⊃RT

2 - p⊃qp
⊃PQ⊃⊃R⊃ST⊃RS
⊃⊃R⊃ST⊃Q⊃RT
connectives identiques

3 - ⊃q⊃PQ
⊃⊃R⊃ST⊃RS
⊃⊃R⊃ST⊃Q⊃RT
on substitue ⊃PQ à p

4 - q⊃PQ
⊃R⊃ST⊃RS
⊃⊃R⊃ST⊃Q⊃RT
connectives identiques

5 - ⊃PQ
⊃RS
⊃⊃R⊃ST⊃Q⊃RT
on substitue ⊃R⊃ST à q.
q n'apparaissant pas ailleurs, on
n'effectue aucune opération.

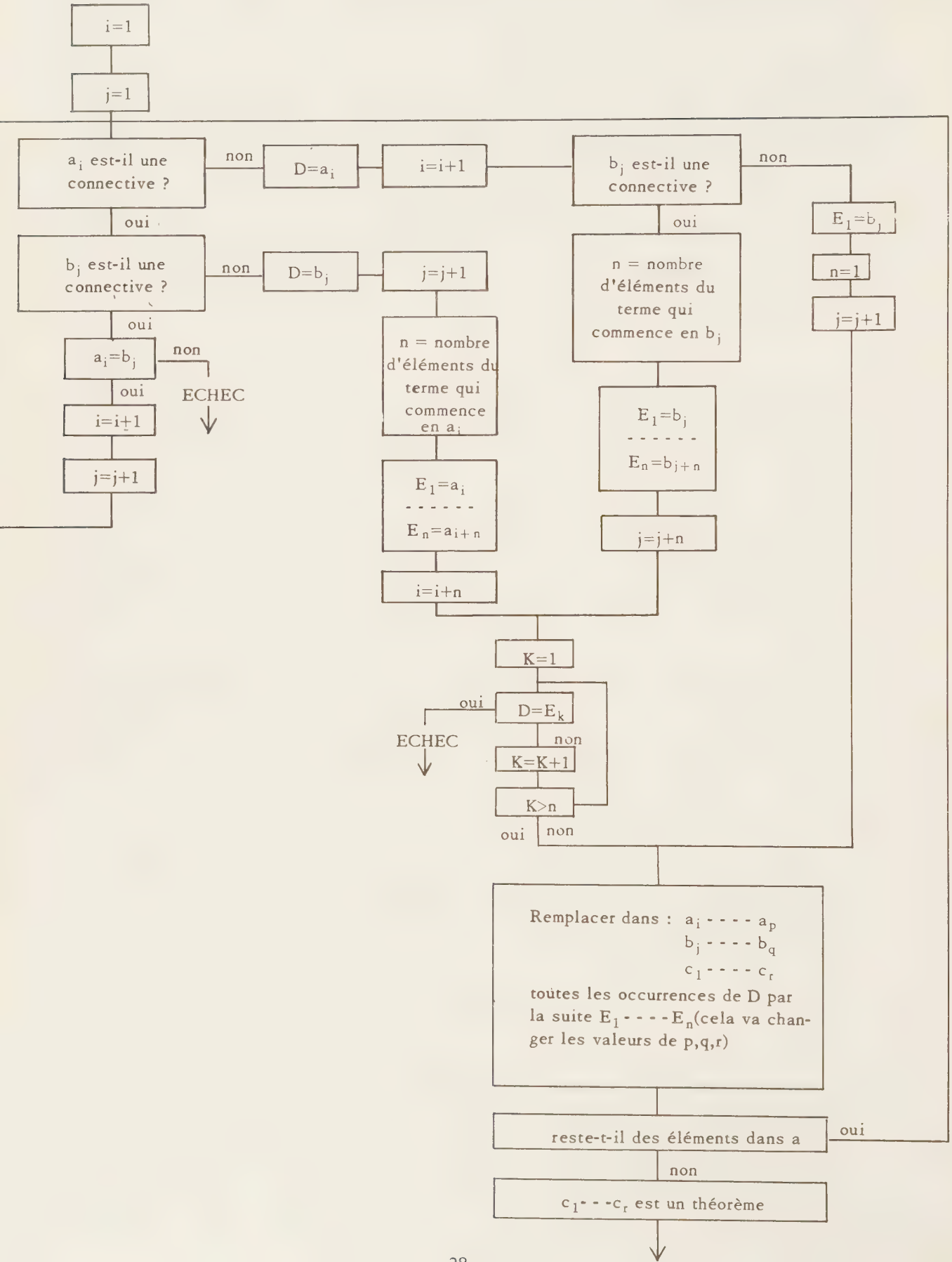
6 - PQ
RS
⊃⊃R⊃ST⊃Q⊃RT
connectives identiques

7 - Q
S
⊃⊃R⊃ST⊃Q⊃RT
on substitue R à P qui n'apparait
nulle part.

8 - Λ
Λ
⊃⊃R⊃ST⊃S⊃RT
on substitue S à Q

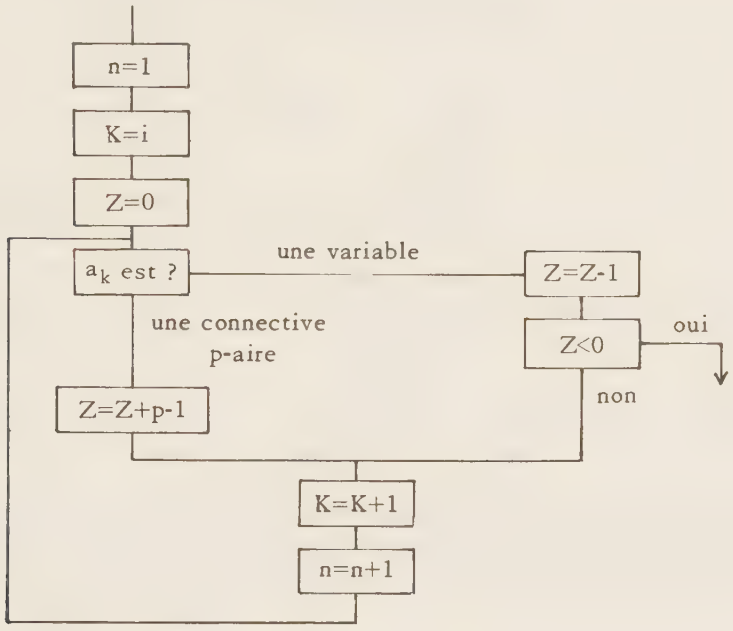
D'où le résultat : ⊃⊃R⊃ST⊃S⊃RT ou ⊃⊃p⊃qr⊃q⊃pr.
et : ⊃p⊃qp+⊃⊃pq⊃⊃r⊃st⊃rs⇒⊃⊃r⊃st⊃q⊃rt⊣⊢⊃⊃p⊃qr⊃q⊃pr.

Donnons un organigramme possible pour cet algorithme.



Nous avons supposé dans l'organigramme que l'on mettait la suite des symboles de a, b et c sous forme de variables indicées. Pratiquement, on a utilisé des listes qui facilitent beaucoup les opérations de substitution et ne demandent pas de recalculer les valeurs de p, q et r. Certains détails n'ont pas été précisés : par exemple la façon dont on s'aperçoit qu'une suite de symboles est terminée.

Détaillons l'organigramme qui calcule le nombre d'éléments du terme commençant en a_i :



Z est le compteur qui a pour valeur le rang (cf le théorème fondamental de la notation polonaise en 2A1b).

Pour des raisons de rapidité d'exécution, la programmation de l'algorithme a été assez différente. On a en particulier compliqué l'organigramme pour avoir la possibilité de faire agir une production non pas sur un théorème isolé, mais sur un ensemble de théorèmes archivés en arborescence.

Il n'en reste pas moins que le problème des dérivations est extrèmement simple à programmer. Les difficultés viennent toutes de la partie heuristique. Le programme a souvent sorti des résultats sans intérêt, mais jamais de résultats qui n'étaient pas des théorèmes de l'axiomatique étudiée (sauf en cas d'erreur machine !)

Remarque 1 - La simplicité de l'algorithme est en grande partie dûe à l'utilisation de la notation polonaise. Il suffit de balayer la suite de symboles de gauche à droite sans avoir à retourner en arrière. On aurait pu établir un algorithme analogue pour la notation suffixée en faisant le balayage à partir de la droite.

Remarque 2 - Il est indispensable que les variables figurant dans a d'une part et celles figurant dans b⇒c d'autre part aient des noms différents : sans cela on risque de ne pas avoir le résultat le plus général ou même un échec alors qu'une dérivation était possible.

Exemple : ⊃p⊃qp+⊃⊃pqr⇒⊃qr↦?

Opérons d'abord sans précautions :

1 - $\supset p \supset qp$
$\supset\supset pqr$
$\supset qr$

2 - $p \supset qp$
$\supset pqr$
$\supset qr$

et l'on a un échec, car on ne peut substituer $\supset pq$ à p.

En réalité, on a :

1 - $\supset p \supset qp$
$\supset\supset PQR$
$\supset QR$

2 - $p \supset qp$
$\supset PQR$
$\supset QR$

3 - $\supset q \supset PQ$
R
$\supset QR$

4 - Λ
Λ
$\supset Q \supset q \supset PQ$

d'où finalement : $\supset p \supset qp + \supset\supset pqr \Rightarrow \supset qr \dashv\vdash \supset p \supset q \supset rp$

Remarque 3 - Le premier cas d'échec (différence des connectives) est très fréquent dans les axiomatiques qui ont plusieurs sortie de connectives. Il ne peut évidemment pas se produire dans celles qui n'ont qu'une seule sorte de connectives, par exemple les axiomatiques 3, 4 et 5.

Exemple : $\supset \wedge pqp + \supset \sim pq \Rightarrow \supset \sim qp \dashv\vdash ?$

1 - $\supset \wedge pqp$
$\supset \sim PQ$
$\supset \sim QP$

2 - $\wedge pqp$
$\sim PQ$
$\supset \sim QP$

et l'on a échec, car $\wedge$ est différent de $\sim$.

Dans certains cas, on pourrait éviter l'échec en utilisant une définition. Si l'on avait la définition $\wedge pq = \sim \supset p \sim q$ on voit que $\supset \wedge pqp$ devient $\supset \sim \supset p \sim qp$ et que l'on a :

$$\supset \sim \supset p \sim qp + \supset \sim pq \Rightarrow \supset \sim qp \dashv\vdash \supset \sim p \supset p \sim q$$

Mais on a renoncé à cette possibilité en considérant comme différents le théorème de départ et celui qui en résulte par application de la définition. Dans l'exemple précédent, si $\supset \wedge pqp$ est issu de $\supset \sim \supset pqp$, on voit que l'on a :

$$\supset \sim \supset pqp + \supset \sim pq \Rightarrow \supset \sim qp \dashv\vdash \supset \sim p \supset pq$$

qui est plus général que $\supset \sim p \supset p \sim q$ puisque l'on peut dériver le second du premier par la substitution : p/ p ;q/ ~q.

Remarque 4 - La deuxième cause d'échec est la seule qui puisse se produire dans les axiomatiques qui ont une seule espèce de connective. Elle a lieu quand on doit substituer à une variable un terme contenant cette variable.

Exemple : $\supset\supset pq \supset\supset rp \supset rq + \supset p \supset\supset qpr \Rightarrow \supset pr \dashv\vdash ?$

1 - $\supset\supset pq \supset\supset rp \supset rq$
$\supset P \supset\supset QPR$
$\supset PR$

2 - $\supset pq \supset\supset rp \supset rq$
$P \supset\supset QPR$
$\supset PR$

3 - $\supset\supset rp \supset rq$
$\supset\supset Q \supset pqR$
$\supset\supset pqR$

4 - ⊃rp⊃rq
⊃Q⊃pqR
⊃⊃pqR

5 - rp⊃rq
Q⊃pqR
⊃⊃pqR

6 - p⊃Qq
⊃pqR
⊃⊃pqR

et l'on à échec, car l'on devrait substituer ⊃pq à p. Il est bien évident qu'aucune substitution ne peut coïncider les suites de symboles p⊃Qq et ⊃pqR.

Remarque 5 - Nous avons traité le cas des dérivations liées au MMthéorème a;a⇒b⟶b auquel nous avons associé l'opération représentée par A+B⇒C↣D. Ceci s'étend aisément à d'autres MMthéorèmes.

1 - a⇒b;b⇒c⟶a⇒c. Si on veut appliquer ce MMthéorème aux productions A⇒B et C⇒D, on aura la production E⇒F obtenue à partir de A et D, quand on fait coïncider B et C par des substitutions.

On applique donc le procédé précédent en mettant B sur la première ligne, C sur la deuxième et A⇒D dans la troisième. Bien entendu au départ aucune variable de A⇒B ne doit figurer dans C⇒D.

On écrira cette dérivation sous la forme :

A⇒B+C⇒D↣E⇒F

étant entendu qu'il n'y a pas commutativité. Si l'on considère C⇒D+A⇒B↣E⇒F
on ferait coïncider A et D et le résultat serait obtenu à partir de C⇒B.

Exemple : ⊃p⊃qr⇒⊃⊃pq⊃pr+⊃⊃pqr⇒⊃qr↣?

1 - ⊃⊃pq⊃pr
⊃⊃PQR
⊃p⊃qr⇒⊃QR

2 - ⊃pq⊃pr
⊃PQR
⊃p⊃qr⇒⊃QR

3 - pq⊃pr
PQR
⊃p⊃qr⇒⊃QR

4 - q⊃pr
QR
⊃p⊃qr⇒⊃QR

5 - ⊃pr
R
⊃p⊃qr⇒⊃qR

6 - Λ
Λ
⊃p⊃qr⇒⊃q⊃pr

Donc : ⊃p⊃qr⇒⊃⊃pq⊃pr+⊃⊃pqr⇒⊃qr↣⊃p⊃qr⇒⊃q⊃pr
et inversement on verrait que : ⊃⊃pqr⇒⊃qr+⊃p⊃qr⇒⊃⊃pq⊃pr↣⊃⊃pq⊃rs⇒⊃⊃qr⊃qs.

2 - a⇒b;b,c⇒d⟶a,c⇒d. On a les productions A⇒B et C,D⇒E. On veut faire coïncider par substitution B et C. Donc on mettra B sur la 1ère ligne, C sur la deuxième et A,D⇒E sur la troisième. On écrira la dérivation : A⇒B+C,D⇒E↣F,G⇒H
Si l'on voulait faire coïncider B et D, on écrirait la deuxième production D,C⇒E aussi on supposera que dans cette écriture la première production agit sur le premier antécédent de la deuxième.

Exemple : p⇒⊃qp+⊃p⊃qr,⊃pq⇒⊃pr↣?

1 - ⊃qp
⊃P⊃QR
p,⊃PQ⇒⊃PR

2 - qp
P⊃QR
p,⊃PQ⇒⊃PR

3 - p
⊃QR
p,⊃qQ⇒⊃qR

4 - Λ
Λ
⊃QR,⊃qQ⇒⊃qR

d'où : $p \Rightarrow \supset qp + \supset p \supset qr, \supset pq \Rightarrow \supset pr \leftrightarrow \supset pq, \supset rp \Rightarrow \supset rq$

On verrait que, par permutation des antécédents, on aurait :

$$p \Rightarrow \supset qp + \supset pq, \supset p \supset qr \Rightarrow \supset pr \leftrightarrow p, \supset q \supset pr \Rightarrow \supset qr$$

3 - $a,b \Rightarrow c ; c \Rightarrow d \longrightarrow a,b \Rightarrow d$. On a les productions $A,B \Rightarrow C$ et $D \Rightarrow E$ et l'on veut faire coïncider par substitutions C et D. On met donc C sur la première ligne, D sur la deuxième et $A,B \Rightarrow E$ sur la troisième. On écrira : $A,B \Rightarrow C + D \Rightarrow E \leftrightarrow F, G \Rightarrow H$

Exemple : $\supset \sim pq, \supset rq \Rightarrow \supset \supset prq + \supset p \supset qr \Rightarrow \supset q \supset pr \leftrightarrow ?$

1 - $\supset \supset prq$
$\supset P \supset QR$
$\supset \sim pq, \supset rq \Rightarrow \supset Q \supset PR$

2 - $\supset prq$
$P \supset QR$
$\supset \sim pq, \supset rq \Rightarrow \supset Q \supset PR$

2 - q
$\supset QR$
$\supset \sim pq, \supset rq \Rightarrow \supset Q \supset \supset prR$

4 - Λ
Λ
$\supset \sim p \supset QR, \supset r \supset QR \Rightarrow \supset Q \supset \supset prR$

d'où : $\supset \sim pq, \supset rq \Rightarrow \supset \supset prq + \supset p \supset qr \Rightarrow \supset q \supset pr \leftrightarrow \supset \sim p \supset qr, \supset s \supset qr \Rightarrow \supset q \supset \supset psr$

Mais la procédure peut aussi s'appliquer à des MMMthéorèmes.

4 - $a \square a \longrightarrow b \Rightarrow c \rightsquigarrow b \Rightarrow c$. On a le théorème A et le MMthéorème $B \longrightarrow C \Rightarrow D$. On veut avoir le Mthéorème $E \Rightarrow F$ le plus général possible en faisant coïncider A et B. On met donc A sur la première ligne, B sur la deuxième et $C \Rightarrow D$ sur la troisième. On écrira : $A + B \longrightarrow C \Rightarrow D \leftrightarrow E \Rightarrow F$

Exemple : $- \supset p \supset qp + \supset pq \longrightarrow p \Rightarrow q \leftrightarrow ?$

1 - $\supset p \supset qp$
$\supset PQ$
$P \Rightarrow Q$

2 - $p \supset qp$
PQ
$P \Rightarrow Q$

3 - $\supset pq$
Q
$p \Rightarrow Q$

4 - Λ
Λ
$p \Rightarrow \supset qp$

et : $\supset p \supset qp + \supset pq \longrightarrow p \Rightarrow q \leftrightarrow p \Rightarrow \supset qp$

Exemple 2 - $||ppp + |p|qr \longrightarrow p \Rightarrow r \leftrightarrow ?$

1 - $||ppp$
$|P|QR$
$P \Rightarrow R$

2 - $|ppp$
$P|QR$
$P \Rightarrow R$

3 - p
$|QR$
$|pp \Rightarrow R$

4 - Λ
Λ
$||QR|QR \Rightarrow R$

et : $||ppp + |p|qr \longrightarrow p \Rightarrow r \leftrightarrow ||pq|pq \Rightarrow q$

5 - a⇒b□a⇒b⟶c,d⇒e⇝c,d⇒e. On a la production A⇒B et le MMthéorème C⇒D⟶E,F⇒G. On veut faire coïncider par substitutions A⇒B et C⇒D. On met donc A⇒B sur la première ligne, C⇒D sur la deuxième et E,F⇒G sur la troisième. On écrira : A⇒B+C⇒D⟶E,F⇒G⟼H,I⇒J

Exemple : ⊃pq⇒⊃⊃rp⊃rq+p⇒⊃qr⟶p,q⇒r⟼?

1 - ⊃pq⇒⊃⊃rp⊃rq
P⇒⊃QR
P,Q⇒R

2 - ⇒⊃⊃rp⊃rq
⇒⊃QR
⊃pq,Q⇒R

3 - ⊃⊃rp⊃rq
⊃QR
⊃pq,Q⇒R

4 - ⊃rp⊃rq
QR
⊃pq,Q⇒R

5 - ⊃rq
R
⊃pq,⊃rp⇒R

6 - Λ
Λ
⊃pq,⊃rp⇒⊃rq

et : ⊃pq⇒⊃⊃rp⊃rq+p⇒⊃qr⟶p,q⇒r⟼⊃pq,⊃rp⇒⊃rq

On voit que l'on opère sur des connectives et des Mconnectives dans ce cas.

Remarque 6 - Avec une légère modification, l'algorithme peut être utilisé pour les démonstrations de conjectures. La méthode utilisée pour démontrer des conjectures est la suivante : on part de l'énoncé de la conjecture et on lui applique les productions à l'envers jusqu'à ce que l'on trouve un théorème déjà démontré. Ceci sera développé en 4D. Soit donc la conjecture A et supposons que l'on ait la production B⇒C. Si, **sans faire des substitutions dans A**, on peut amener C à coïncider avec A et si B ainsi modifié est un théorème, on aura démontré A. Si l'on faisait des substitutions dans A, on n'aurait pas la démonstration de A, mais celle d'un résultat plus faible. On rajoute ainsi une troisième cause d'échec.

Il suffit de vérifier qu'au cours des substitutions, il n'y en ait aucune où l'on ait à substituer un terme à une variable figurant dans A et d'ajouter ce test à l'organigramme.

En dehors de cela, on applique la méthode normale en mettant A sur la première ligne, C sur la deuxième et B sur la troisième. On l'écrit :

A+B⇒C⟼⟼D

en indiquant par le signe⟼⟼ qu'il s'agit de l'algorithme modifié et que l'on fait coïncider A avec C et non avec B.

Exemple 1 - On veut démontrer ⊃p⊃⊃pqq. On connait la production ⊃p⊃qr⇒⊃q⊃pr et le théorème ⊃pp.

On va évaluer : ⊃p⊃⊃pqq+⊃p⊃qr⇒⊃q⊃pr⟼⟼?

1 - ⊃p⊃⊃pqq
⊃P⊃QR
⊃Q⊃PR

2 - p⊃⊃pqq
P⊃QR
⊃Q⊃PR

3 - ⊃⊃pqq
⊃QR
⊃Q⊃pR

4 - ⊃pqq
QR
⊃Q⊃pR

5 - q
R
⊃⊃pq⊃pR

6 - Λ
Λ
⊃⊃pq⊃pq

et comme ⊃⊃pq⊃pq est un théorème se déduisant de ⊃pp par la substitution p/⊃pq, on a démontré notre conjecture.

Exemple 2 - ⊃p⊃qp+⊃p⊃qr⇒⊃⊃pq⊃pr⟼⟼?

1 - ⊃p⊃qp
⊃⊃PQ⊃PR
⊃P⊃QR

2 - p⊃qp
⊃PQ⊃PR
⊃P⊃QR

et il y a échec, car on devrait substituer à la variable p le terme ⊃PQ. Il est du reste évident que l'on ne peut démontrer ⊃p⊃qp à partir de cette production. On pourrait obtenir par contre le théorème ⊃⊃pq⊃p⊃pq qui est plus faible que ⊃p⊃qp si l'on avait déjà démontré ⊃p⊃q⊃pq :

$$\supset\supset pq\supset p\supset pq \vdash \supset p\supset qr \Rightarrow \supset\supset pq\supset pr \twoheadrightarrow \supset p\supset q\supset pq$$

On pourrait faire une étude analogue avec les productions à deux antécédents : si A est une conjecture et B,C⇒D une production, si l'on peut faire coïncider A et D sans faire de substitutions dans A, A sera démontré si les énoncés transformés de B et de C par les substitutions sont des théorèmes.

On l'écrira : A+B,C⇒D ↠ E,F

Rappelons les conventions que nous avons prises pour représenter l'opération de l'algorithme de dérivation :

- MMthéorème

a;a⇒b ⟶ b	A+B⇒C ↦ D
a⇒b;b⇒c ⟶ a⇒c	A⇒B+C⇒D ↦ E⇒F
a⇒b;b,c⇒d ⟶ a,c⇒d	A⇒B+C,D⇒E ↦ F,G⇒H
a,b⇒c;c⇒d ⟶ a,b⇒d	A,B⇒C+D⇒E ↦ F,G⇒H

- MMMthéorème

a□a ⟶ b⇒c ⇝ b⇒c	A+B ⟶ C⇒D ↦ E⇒F
a⇒b□a⇒b ⟶ c,d⇒e ⇝ c,d⇒e	A⇒B+C⇒D ⟶ E,F = G ↦ H,I⇒J

- Conjectures

- Si a est une conjecture et b⇒a une production, b est une conjecture	A+B⇒C ↠ D
- Si a est une conjecture et b,c⇒a une production, b et c sont des conjectures	A+B,C⇒D ↠ E,F

Par la suite nous ne ferons plus figurer les étapes de l'algorithme et nous écrirons directement le résultat après les signes ↦ et ↠. S'il y a échec, on écrit "Echec".

2C - ETUDE GENERALE DES DEMONSTRATIONS

2C1 - Formules décrivant une démonstration

- A chaque axiome on associe un nom désigné par une lettre minuscule : a, b, c..... Dans l'axiomatique 1 ⊃⊃p⊃qr⊃⊃pq⊃pr a pour nom a et ⊃p⊃qp : b.

- A chaque production à un antécédent on associe un nom désigné par une lettre majuscule précédant R : A, B, C....

Par définition si "i" est la formule qui représente la démonstration du théorème que l'on fera coïncider avec l'antécédent, Ai sera la formule du résultat de la dérivation. Il est souvent commode de ne pas expliciter la formule qui vient à la place de i.

Exemple : si A est ⊃⊃pqr⇒⊃qr, i est ⊃⊃pqr et Ai est ⊃qr.

Si a est ⊃⊃p⊃qr⊃⊃pq⊃pr, Aa est une formule de démonstration de ⊃⊃pq⊃⊃rp⊃rq.

- A chaque production à deux antécédents on associe un nom désigné par une lettre majuscule suivant R : R, S, T...

Si i est la formule de démonstration du théorème qui va coïncider avec le premier antécédent et j avec le deuxième, Rij sera la formule du résultat.

Exemple : si R est p,⊃pq⇒q, i est p,j est ⊃pq et Rij est q.

Si on ne précise qu'un des antécédents, on a une production à un antécédent : si b est ⊃p⊃qp, i est p et Rib : ⊃qp. De même j est ⊃⊃p⊃qpr et Rbj est r.

On peut préciser les deux antécédents :

Rbb est la formule d'une démonstration de ⊃p⊃q⊃rq et Rba de ⊃⊃pq⊃pp.

D'une manière générale, si on applique une production à un antécédent de formule $\alpha i \beta$ (les lettres grecques représentent des suites de symboles qui peuvent être vides où figurent les noms des axiomes et des productions de l'axiomatique) à un théorème de formule γ, le résultat sera un théorème de formule $\alpha\gamma\beta$.

Si de même on applique une production à deux antécédents de formule $\alpha i\beta j\gamma$ (Resp.$\alpha j\,\beta i\,\gamma$) à deux théorèmes de formule : δ pour celui qui coïncidera avec le premier antécédent et θ pour l'autre, le résultat sera un théorème de formule $\alpha\delta\beta\theta\gamma$ (resp. $\alpha\theta\beta\delta\gamma$).

Remarque 1 - Il faut distinguer avec soin énoncé et formule de démonstration. L'énoncé est la suite de symboles qui représente un résultat, la formule de démonstration donne le moyen de trouver ce résultat dans une certaine axiomatique. Remarquons :

- à un énoncé peut correspondre plusieurs formules de démonstrations : il y a alors ambiguité. Ceci est très fréquent et l'on verra même que pour certains énoncés il se peut qu'il y corresponde une infinité de formules de démonstration.
- à une formule de démonstration correspond un résultat de base obtenu par des moyens analogues à ceux décrits en 2B. Il y a aussi tous les résultats qui peuvent s'en déduire par substitution. Nous nous intéresserons uniquement au résultat le plus fort et on peut alors dire qu'à une formule de démonstration correspond au plus un énoncé.

Il se peut qu'aucun énoncé ne corresponde à une formule, car il peut y avoir échec au cours des dérivations.

Exemple : Si a est ~∧p~p et A : ⊃⊃pqr⇒⊃qr, à Aa ne correspondra aucun théorème, car :

$$\sim\wedge p\sim p + \supset\supset pqr \Rightarrow \supset qr \longrightarrow \text{Echec.}$$

Remarque 2 - A un théorème correspond une formule de démonstration ne contenant ni i ni j. Une production à un antécédent a une occurrence de i et une production à deux antécédents une occurrence de i et une de j.

Un MMthéorème n'a pas une formule de démonstration, mais il peut lui correspondre une loi de transformation des formules de démonstration. Voyons cela un peu plus en détail. Si l'on a la production A,B⇒C de formule Rij, on voit que si A est un théorème, B⇒C est une production, donc on a le MMthéorème $A \longrightarrow B \Rightarrow C$. Mais si A a pour formule α,B⇒C aura pour formule Rαj (ou Rαi) et l'on voit que si α est la formule d'un théorème, après application du MMthéorème on aura une

production de formule Rαi. On écrit cette action de transformation des MMthéorèmes sous la forme $\alpha \rightarrow f(\alpha)$, f étant une fonction caractéristique du MMthéorème. Dans l'exemple, on a $\alpha \rightarrow R\alpha i$.

Remarque 3 - Le nom que l'on donne aux deux antécédents d'une production est purement conventionnel. On peut choisir n'importe quel ordre, mais il faut s'y tenir. i correspondra au premier antécédent tel qu'il figure dans l'énoncé de la production et j au deuxième.

Remarque 4 - Il est commode de définir de nouveaux noms pour éviter d'avoir des formules trop longues.

Exemple 1 - Si dans l'axiomatique 1 on pose d=Rbb, on peut écrire :

RRbbRRRaRRbbaRRabaa sous la forme :

RdRRRaRdaRRabaa

Exemple 2 - Toujours dans la même axiomatique si l'on pose :

Ai pour RbRRiba (qui correspond à l'énoncé ⊃⊃pqr⇒⊃qr)

Bi pour ARia (qui correspond à ⊃p⊃qr⇒⊃q⊃pr) on peut écrire :

RbRRRRbRRRbRRRaababaaba sous la forme :

BABa

On voit que cela amène automatiquement à considérer des Mthéorèmes nouveaux.

Remarque 5 - On a défini ainsi un système formel utilisant la notation polonaise, les noms des théomes étant des constantes, i et j représentant des variables, les productions à p antécédents jouant le rôle de connectives p-aires.

Algorithme permettant d'obtenir l'énoncé à partir de la formule de démonstration

Il est important quand on a une formule de démonstration de chercher à quel énoncé elle correspond (s'il y en a un, car il peut y avoir échec). Nous allons donner l'énoncé de l'algorithme, puis des exemples et enfin nous donnerons la justification de l'algorithme.

Enoncé. On veut découvrir à quel énoncé correspond Rαβ, α et β étant des termes, et l'on sait à quel énoncé ils correspondent. Ils peuvent aussi bien être des théorèmes que des productions à 1 ou 2 antécédents.

On utilise 5 lignes

1 - on met à la suite les deux antécédents de la production R.

2 - on met à la suite les énoncés des théorèmes correspondants à α et β ou, si ce sont des productions, les résultats de ces productions.

3 - on met l'antécédent correspondant à i s'il apparait dans α ou β.

4 - on met l'antécédent correspondant à j, s'il apparait.

5 - on met le résultat de la production R.

Par le procédé indiqué en 2B, on va amener par des substitutions les suites des deux premières

lignes à coïncider. On fera les substitutions dans les trois dernières lignes et dans les queues des deux premières. Quand on aura terminé l'énoncé correspondant à $R\alpha\beta$ sera dans la ligne 5 si c'est un théorème ou, si c'est une production, aura ses antécédents dans les lignes 3 et 4 et son résultat dans la ligne 5.

Il se peut qu'il y ait échec au cours des substitutions pour une des deux causes vues en 2B. A ce moment il n'y a aucun énoncé qui correspond à la formule.

Naturellement, il faut donner des noms différents aux variables qui apparaissent dans l'énoncé correspondant à α, dans celui correspondant à β et dans celui de la production R.

Si l'on veut voir à quel énoncé correspond $A\alpha$, on opère de manière analogue en mettant sur la première ligne l'antécédent de A et sur la deuxième l'énoncé correspondant à α si c'est un théorème ou le résultat si c'est une production.

Quand on a une formule de démonstration, on va voir à quel énoncé elle correspond en appliquant le procédé précédent d'abord aux termes du type $R\alpha\beta$ ou $A\alpha$ où α et β sont des noms d'axiomes, puis on reconnaitra de proche en proche les énoncés correspondants à des termes de plus en plus importants.

Dans le cas où on a affaire à un théorème, on supprime les lignes 3 et 4 qui sont inutiles. Pour une production à un antécédent, on supprime la ligne 4.

Exemple 1 - Soit dans l'axiomatique 1, la formule RbRRaba

- Rab. Pour une fois, donnons les intermédiaires des substitutions

1 - P⊃PQ
⊃⊃p⊃qr⊃⊃pq⊃pr⊃u⊃vu
Q

2 - ⊃⊃⊃p⊃qr⊃⊃pq⊃prQ
⊃u⊃vu
Q

3 - ⊃⊃p⊃qr⊃⊃pq⊃prQ
u⊃vu
Q

4 - Q
⊃v⊃⊃p⊃qr⊃⊃pq⊃pr
Q

5 - Λ
Λ
⊃v⊃⊃p⊃qr⊃⊃pq⊃pr

donc l'énoncé correspondant à Rab est ⊃p⊃⊃q⊃rs⊃⊃qr⊃qs

- RRaba

P⊃PQ
⊃p⊃⊃q⊃rs⊃⊃qr⊃qs⊃⊃u⊃vw⊃⊃uv⊃uw
Q

et après substitutions : ⊃⊃p⊃q⊃rs⊃p⊃⊃qr⊃qs

- RbRRaba

P⊃PQ
⊃p⊃qp⊃⊃u⊃v⊃wx⊃u⊃⊃vw⊃vx
Q

et finalement on a l'énoncé cherché : ⊃⊃pq⊃⊃rp⊃rq

On a utilisé seulement trois lignes, car on savait que l'on avait un théorème : dans la formule ne figurent en effet ni i ni j.

Exemple 2 - Toujours dans l'axiomatique 1 :

RRbbRRRaRRbbaRRabaa

Nous n'expliciterons pas les tableaux, nous nous contenterons d'indiquer les énoncés correspondants aux différents termes, en séparant par des points les deux termes auxquels on applique la production R

R.b.b : ⊃p⊃q⊃rq
R.Rbb.a : ⊃⊃pq⊃p⊃rq
R.a.b : ⊃p⊃⊃q⊃rs⊃⊃qr⊃qs
R.Rab.a : ⊃⊃p⊃q⊃rs⊃p⊃⊃qr⊃qs
R.a.RRbba : ⊃⊃p⊃qr⊃s⊃⊃pq⊃pr
R.RaRRbba.RRaba : ⊃⊃p⊃qr⊃⊃s⊃pq⊃s⊃pr
R.RRaRRbbaRRaba.a : ⊃⊃⊃p⊃qr⊃s⊃pq⊃⊃p⊃qr⊃s⊃pr
R.Rbb.RRRaRRbbaRRabaa : ⊃⊃p⊃qr⊃q⊃pr

Exemple 3 - Toujours dans l'axiomatique 1 : RbRRiba

On voit que l'on a une production à un antécédent

- Rib

1 - P⊃PQ
i⊃p⊃qp
i
Q

2 - ⊃PQ
⊃p⊃qp
P
Q

3 - PQ
p⊃qp
P
Q

4 - Q
⊃qP
P
Q

5 - Λ
Λ
P
⊃qP

et l'on a donc la production p⇒⊃qp

- RRiba. On a le tableau :

P⊃PQ
⊃qp⊃⊃u⊃vw⊃⊃uv⊃uw
p
Q

et après substitutions : ⊃pq⇒⊃⊃rp⊃rq

- RbRRiba : P⊃PQ
⊃p⊃qp⊃⊃wu⊃wv
⊃uv
Q

et finalement l'énoncé cherché : ⊃⊃pqr⇒⊃qr.

Exemple 4 - Toujours dans l'axiomatique 1 : RRjbRia. C'est une production à deux antécédents. Il nous faudra donc 5 lignes.

- Ria P⊃PQ
  i⊃⊃p⊃qr⊃⊃pq⊃pr
  i
  Λ
  Q

et la production : ⊃p⊃qr⇒⊃⊃pq⊃pr

- Rjb : P⊃PQ
  j⊃p⊃qp
  Λ
  j
  Q

et la production p⇒⊃qp

- RRjbRia : P⊃PQ
  ⊃qp⊃⊃uv⊃uw
  ⊃u⊃vw
  p
  Q

et finalement l'énoncé cherché est : ⊃p⊃qr,q⇒⊃pr

(attention à l'ordre des antécédents. RRibRja serait p,⊃q⊃pr⇒⊃qr)

Justification de l'algorithme

Il y a un grand nombre de cas particuliers qui se traitent tous de manière analogue. Supposons que l'on cherche l'énoncé correspondant à Rαβ sachant que α correspond à la production à un antécédent A⇒B et β au théorème C.

Cela signifie que l'on a appliqué la production R en faisant coïncider B avec le premier antécédent et C avec le deuxième. Une fois les substitutions faites, on aura :

A'⇒B' à la place de A⇒B
C' à la place de C
B',C'⇒F' à la place de D,E⇒F énoncé de R

Si A' est un théorème, alors B' est un théorème et comme C' est un théorème, F' est un théorème. On voit que l'on a la démonstration de la production A'⇒F' qui correspond à Rαβ.

L'algorithme proposé a bien pour effet de faire coïncider les énoncés correspondants à α et β avec les antécédents de R en faisant les substitutions dans l'antécédent de α et dans le résultat de R.

D'une manière générale les résultats des productions se traitent de la même façon que les théorèmes, alors que les antécédents deviennent des antécédents pour la production finale.

L'obligation d'avoir des variables ayant des noms différents dans A⇒B, C et D,E⇒F vient de ce que l'on veut ne faire que les substitutions strictement indispensables pour avoir l'énoncé le plus fort possible, tous les autres s'en déduisant par substitution.

Remarque. Quand on a la formule correspondant à un théorème, on peut substituer i à un terme quelconque qui y figure et l'on a alors une décomposition de la forme :

$$A+B\Rightarrow C \dashv\vdash D$$

Si D a pour formule $\alpha\beta\gamma$, β étant un terme au sens de la notation polonaise, A a pour formule β et $B\Rightarrow C$: $\alpha i\gamma$

Il y a un grand nombre de décompositions possibles.

Exemple : soit dans l'axiomatique 1 : RaRRRRbbRRbRRabaaba qui est une formule correspondant à l'énoncé : CCpCqrCqCpr

1 - β est a ; $\alpha i\gamma$ est RiRRRRbbRRbRRabaaba. Cela correspond à :

CCpCqrCCpqCpr+CpCqrs⇒CpCrs⊣⊢CCpCqrCqCpr

2 - β est RRRbbRRbRRabaab ; $\alpha i\gamma$ est RaRia. Cela correspond à :

CpCCCqrCrs+CCpCqrCCCpqCprs⇒CCpCqrs⊣⊢CCpCqrCqCpr

3 - β est RRbbRRbRRabaa ; $\alpha i\gamma$ est RaRRiba. Cela correspond à :

CCCpqrCqr+CCCpqCprs⇒CCpCqrs⊣⊢CCpCqrCqCpr

4 - β est Rbb ; $\alpha i\gamma$ est RaRRRiRRbRRabaaba. Cela correspond à :

CpCqCrq+CCCpqCprCsCpq⇒CCpCqrCsCpr⊣⊢CCpCqrCqCpr

5 - β est b ; $\alpha i\gamma$ est RaRRRRibRRbRRabaaba. Cela correspond à :

CpCqp+CpCqr⇒CCqCrsCpCqs⊣⊢CCpCqrCqCpr

6 - β est RbRRaba ; $\alpha i\gamma$ est RaRRRRbbRiaba. Cela correspond à :

CCpqCCrpCrq+CCCpqCprCCsCtsu⇒CCpCqru⊣⊢CCpCqrCqCpr

7 - β est b ; $\alpha i\gamma$ est RaRRRRbbRRiRRabaaba. Cela correspond à :

CpCqp+CCCpqCprCsCCtsu⇒CCpCqrCsu⊣⊢CCpCqrCqCpr

8 - β est Rab ; $\alpha i\gamma$ est RaRRRRbbRRbRiaaba. Cela correspond à :

CpCCqCrsCCqrCqs+CCpqCprCCsCCpqCprCCtCutv⇒CCpCqrv⊣⊢CCpCqrCqCpr

9 - β est a ; $\alpha i\gamma$ est RaRRRRbbRRbRRibaaba. Cela correspond à :

CCpCqrCCpqCpr+CCpCCqrCqsCCtCutv⇒CCpCqrv⊣⊢CCpCqrCqCpr

10 - β est b ; $\alpha i\gamma$ est RaRRRRbbRRbRRaiaaba. Cela correspond à :

CpCqp+CCCpCqrCCpqCprCCCstCsuCCvCCstCsuCCwCxwy⇒CCsCtuy⊣⊢CCpCqrCqCpr

On pourrait donner encore d'autres décompositions de la même formule. Il y en a 2p pour une formule contenant p fois la lettre R. La plupart sont totalement dénuées d'intérêt.

Cet exemple permet de voir comme il est facile de présenter une démonstration d'une façon différente de sa découverte. Heuristiquement, la démonstration précédente a été ainsi trouvée : on voulait démontrer CCpCqrCqCpr. Le résultat le plus proche que l'on connaissait étant l'axiome a : CCpCqrCCpqCpr, on a vu qu'il suffisait d'enlever la deuxième occurrence de p, donc de démontrer la production CpCCqrs⇒CpCrs. Comme on avait le MMthéorème Cpq⊣⊢Crp⇒Crq il suffisait d'avoir le théorème CCCpqrCqr qui est plus facile à démontrer

Une fois la démonstration trouvée, on a facilement la formule de démonstration et on peut la présenter sous un aspect très différent, par exemple sous la dixième forme ! Cet exemple est évidemment caricatural, mais il est quand même regrettable que la présentation d'une solution soit souvent sans aucun rapport avec les heuristiques qui ont amené sa découverte.

Le programme n'utilise pas les formules de démonstration, cela aurait été possible, mais l'on manquait de la place en mémoire pour les stocker. Il aurait fallu aussi programmer les algorithmes qui les calculent. D'autre part le programme n'avait pas les moyens méthathéoriques (comme le théorème de déduction) qui les utilisent et cela aurait perdu beaucoup d'intérêt.

Par la suite nous les utiliserons pour indiquer avec précision les démonstrations. C'est une forme très condensée d'exposition puisqu'en une ligne on indique un grand nombre de dérivations C'est, au cours d'une démonstration, un fil conducteur très sûr : cela oblige à expliciter toutes les opérations que l'on fait.

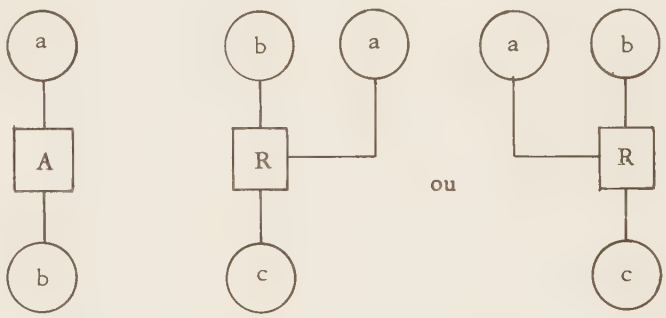
2C2 - Représentation graphique des démonstrations

Nous allons définir un procédé isomorphe du précédent. Mais, étant graphique, il a l'avantage d'être plus parlant.

- A un axiome on fait correspondre son nom à l'intérieur d'un rond.

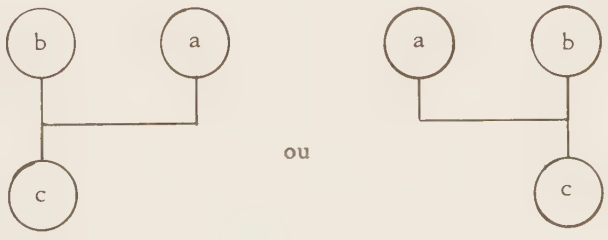
- A une production unaire Ai qui, appliquée au théorème a, donne le résultat Aa, on fait correspondre le schéma :

- A une production à deux antécédents Rij qui, appliquée aux théorèmes a et b, donne le résultat c=Rab on fait correspondre un des schémas ci-contre qui marquent bien le rôle dissymétrique des antécédents.



Il est bien évident que l'on peut par ce procédé représenter graphiquement toute formule de démonstration. On commence par représenter les éléments du type Rab ou Aa, a et b étant des axiomes et les résultats de ces dérivations servent à leur tour d'antécédents pour les productions que l'on n'a pas encore représentées.

Remarque - Dans le cas, très fréquent où il y a une seule production et où elle a deux antécédents, il est inutile de la faire figurer explicitement et on utilise la représentation ci-contre.



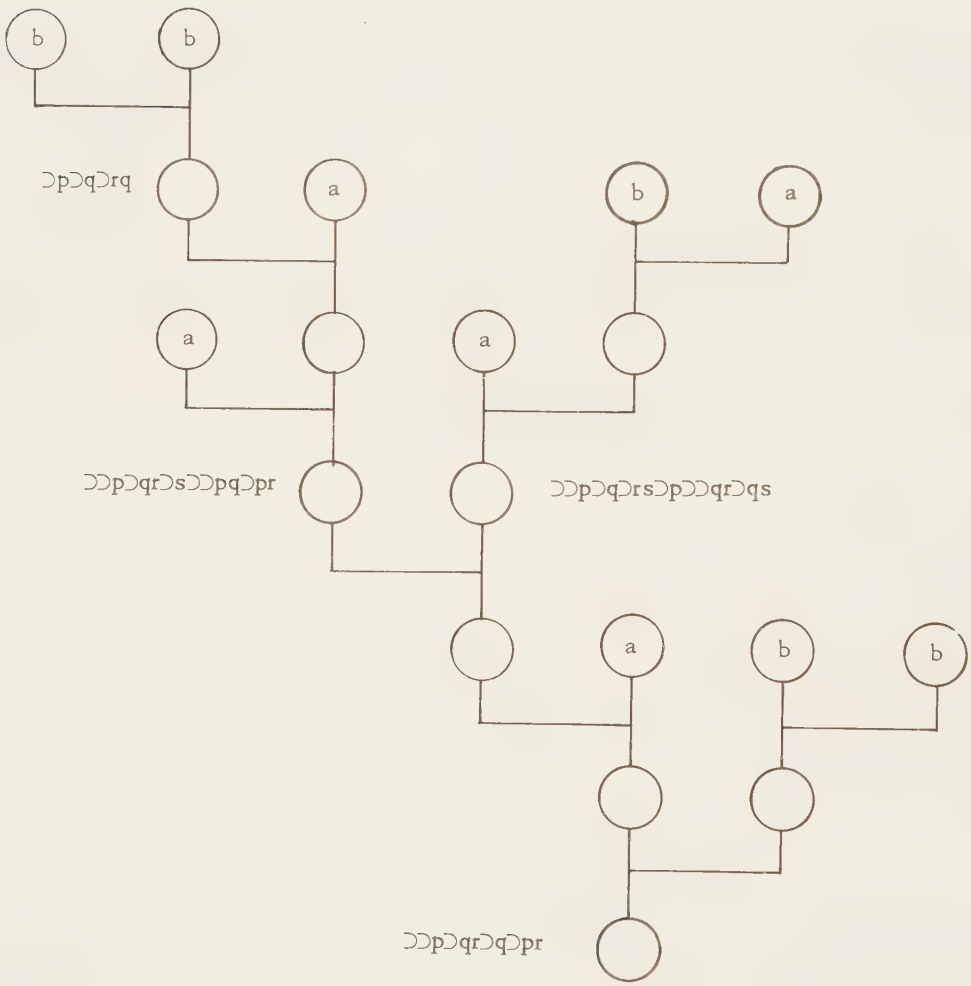
Exemples - On se place dans l'axiomatique 1.

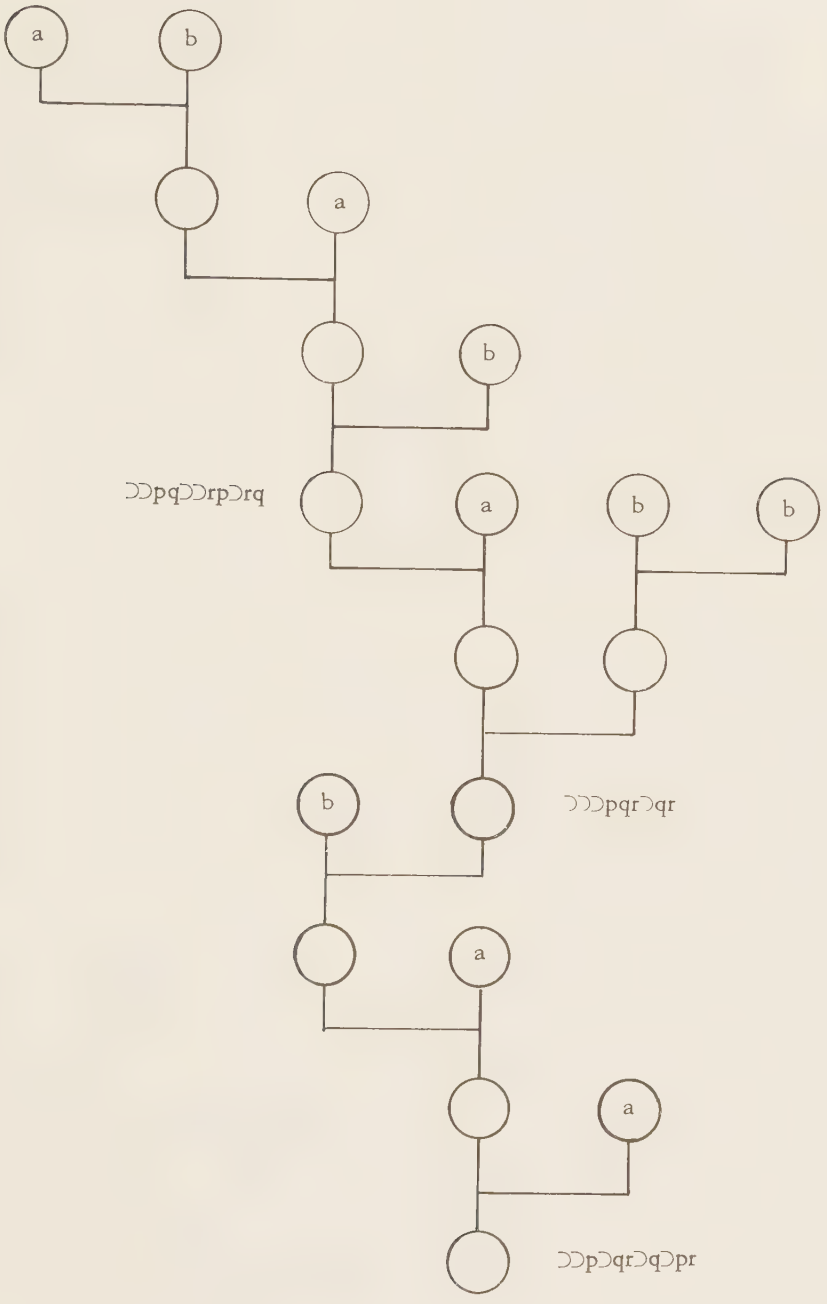
$$\supset\supset p\supset qr\supset\supset pq\supset pr:a \;;\; \supset p\supset qp:b \;;\; p,\supset pq\Rightarrow q:Rij$$

Exemples : on se place dans l'axiomatique 1

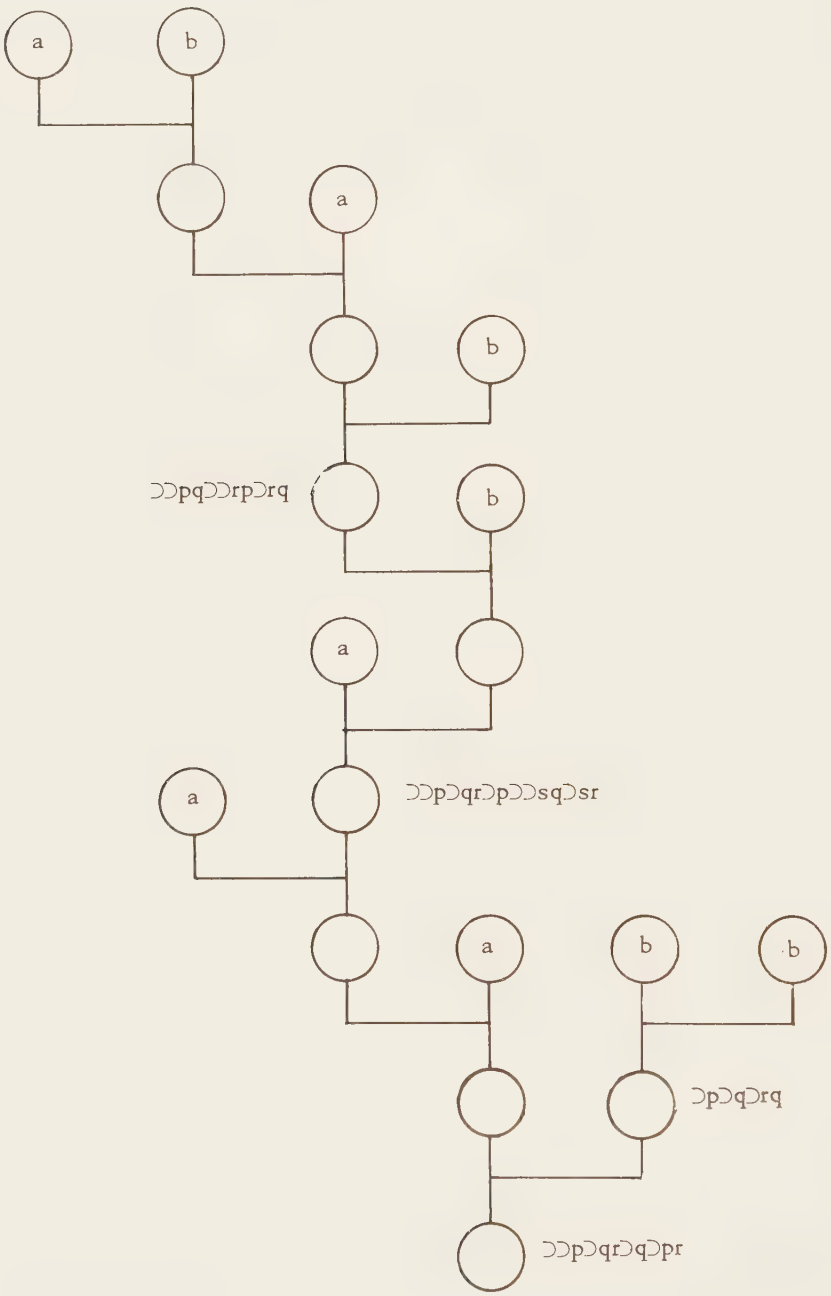
$CCpCqrCCpqCpr$: a ; $CpCqp$: b ; $p,Cpq \Rightarrow q$: Rij.

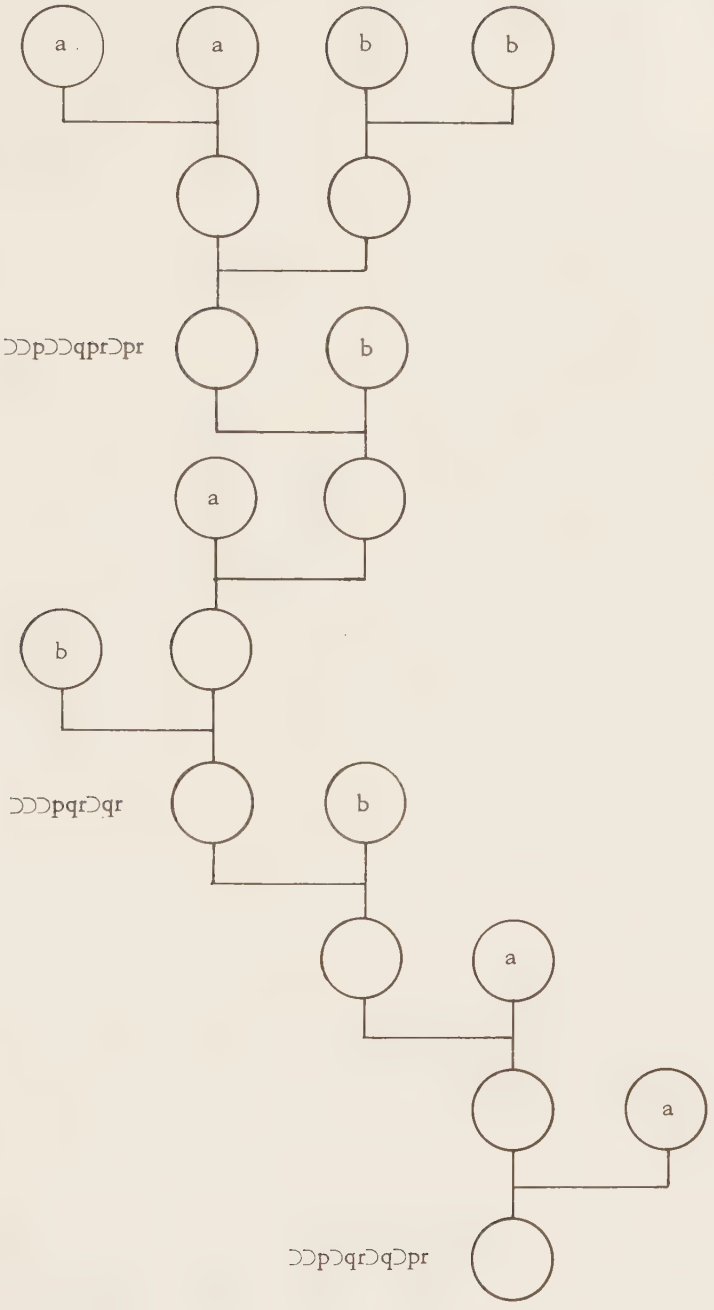
1) RRbbRRRaRRbbaRRabaa. On a déjà donné les intermédiaires en 2C1.




a
b
a
b
CCpqCCrpCrq
a
b
b
b
CCCpqrCqr
a
a
CCpCqrCqCpr

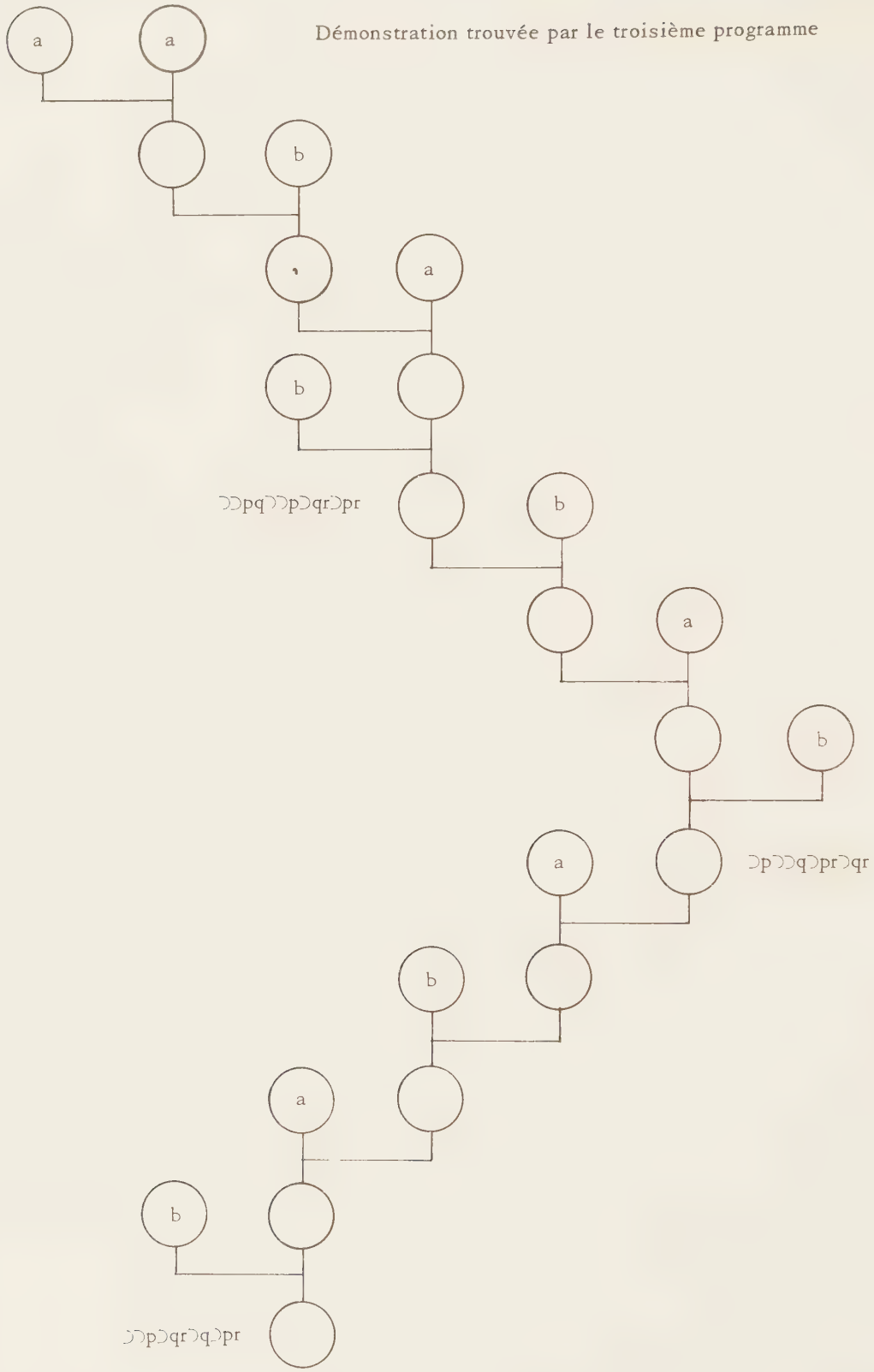
3 - RRbbRRaRRRbRRababaa



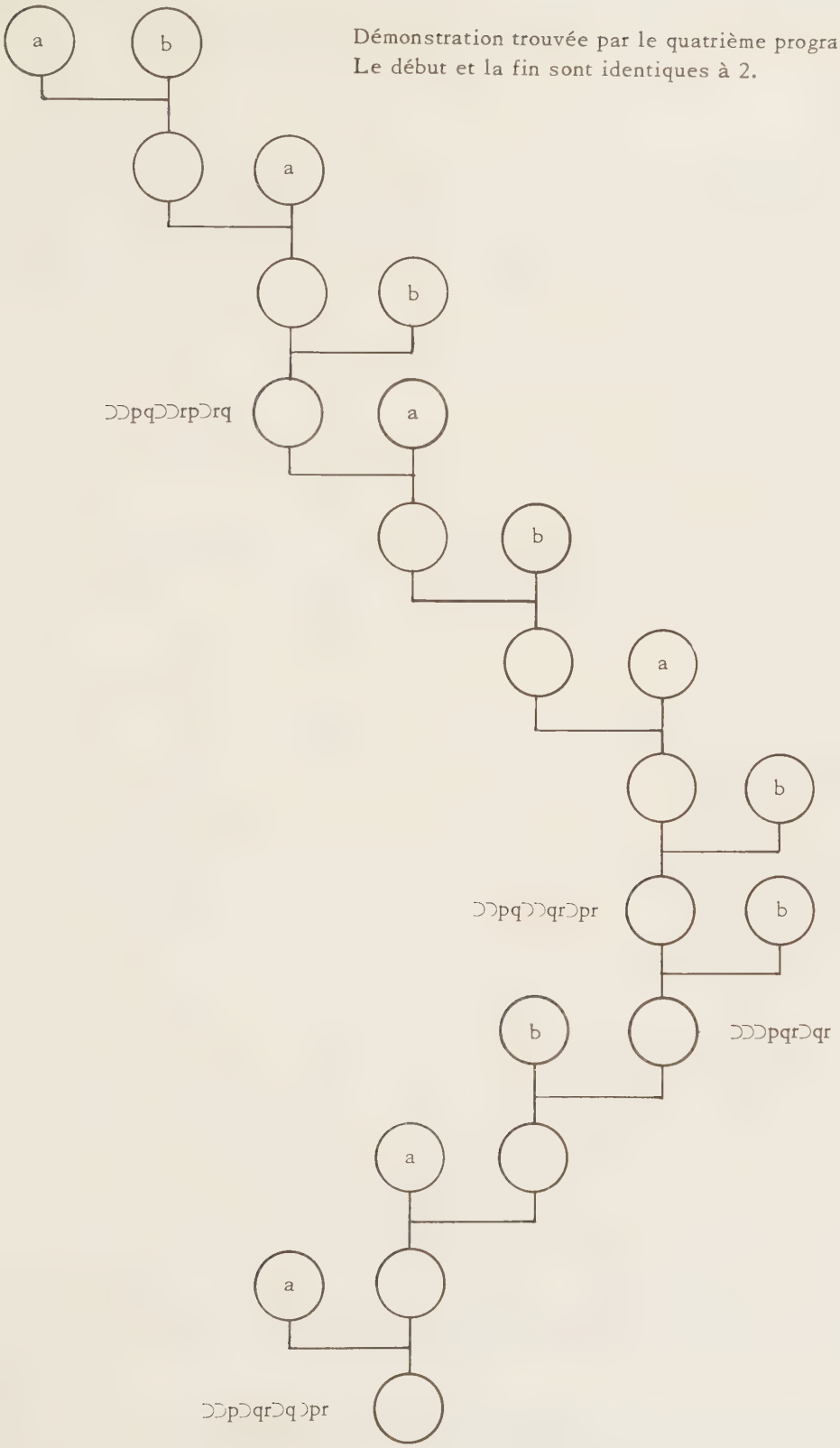

a
a
b
b
⊃⊃p⊃⊃qpr⊃pr
b
a
b
⊃⊃⊃pqr⊃qr
b
a
a
⊃⊃p⊃qr⊃q⊃pr

5 - RbRRRRbRRRbRRRaababaaba

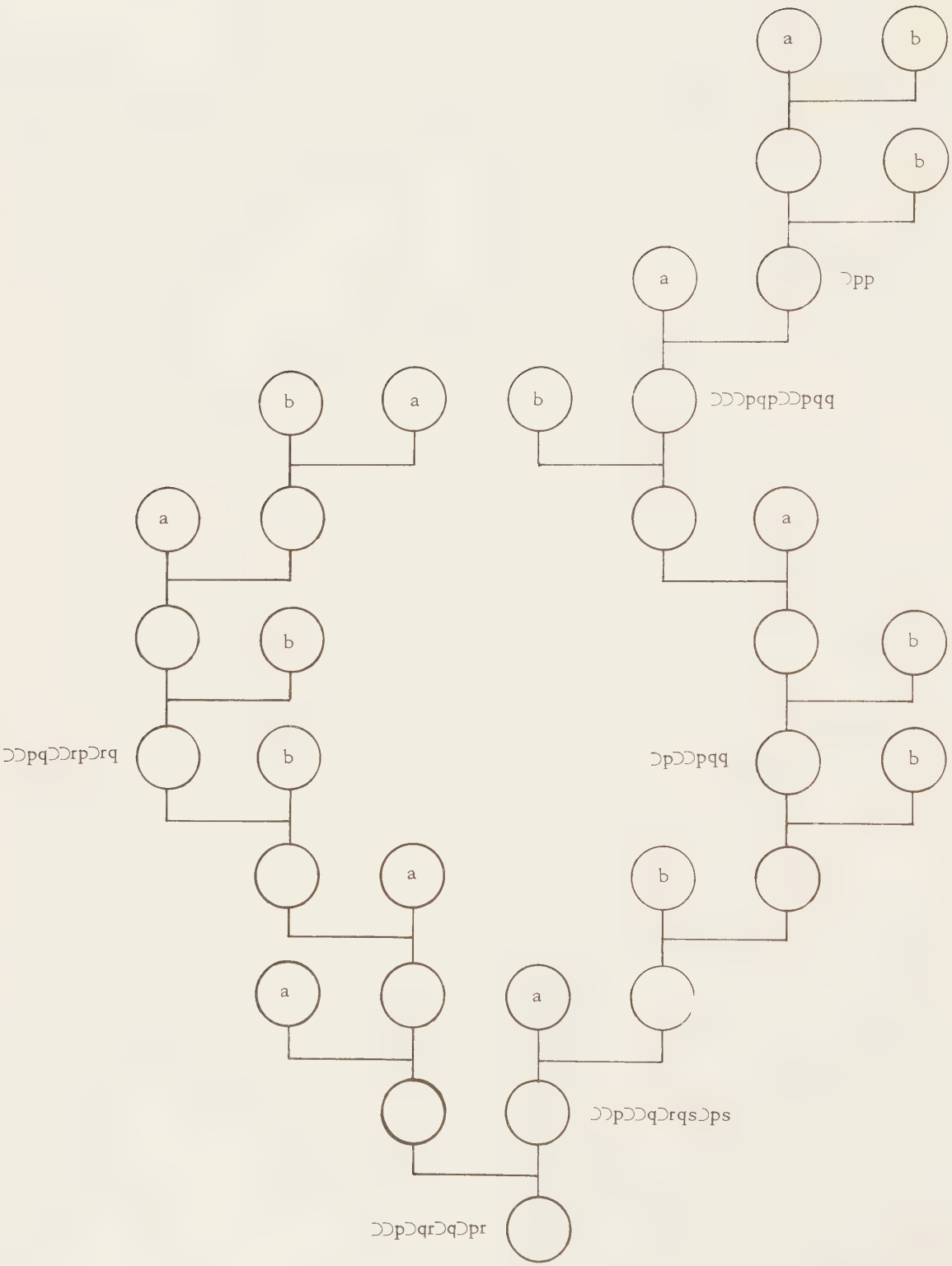
Démonstration trouvée par le troisième programme



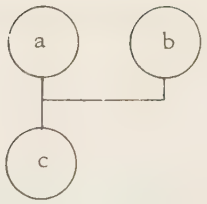
6 - RaRRRbRbRRRRbRRabaababa



Démonstration trouvée par le quatrième programme.
Le début et la fin sont identiques à 2.


a
b
b
a
Cpp
b
a
CCCpqpCCpqq
a
b
b
CCpqCCrpCrq
b
CpCCpqq
b
a
b
a
a
CCpCCqCrqsCps
CCpCqrCqCpr

Remarque - Il est très facile de passer du graphe à la formule de démonstration. On détermine de proche en proche les formules des divers intermédiaires en commençant par le haut. Dans l'exemple ci-contre, si a et b ont pour formules α et β, c aura pour formule $R\beta\alpha$.



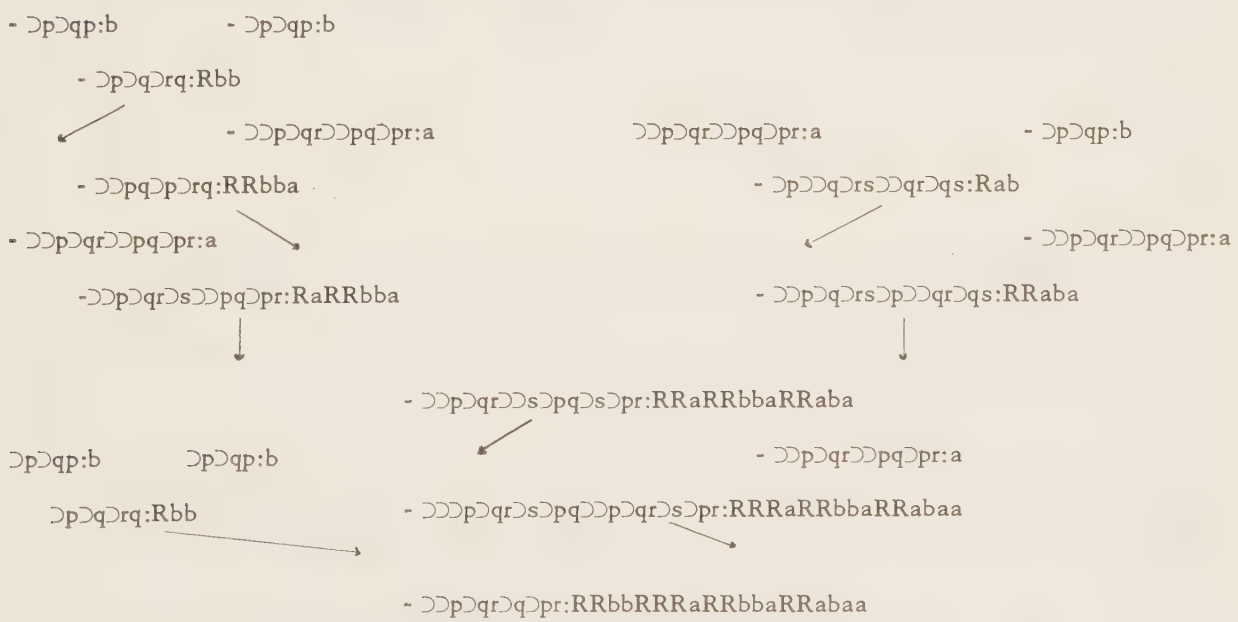
Il existe une autre présentation très proche de la précédente qui met mieux en évidence les intermédiaires. Supposons que l'on ait une seule production et qu'elle ait deux antécédents. Nous mettons sur la première ligne à gauche le théorème que l'on fera coïncider avec le premier antécédent, à droite celui que l'on fera coïncider avec le second et sur la deuxième ligne au milieu le résultat de la dérivation.

Si le résultat est utilisé dans la dérivation suivante, il est commode de l'indiquer par une flèche, sans le réécrire.

Cette notation est particulièrement commode pour les démonstrations dont la représentation graphique est assez linéaire.

Exemple - Reprenons la première démonstration de ↄↄpↄqrↄqↄpr.

Nous ferons suivre l'énoncé de chaque résultat par sa formule de démonstration.



2C3 - Nombres qui caractérisent la difficulté d'une démonstration

Il peut être intéressant de mesurer par des nombres la difficulté d'une démonstration. On peut comparer ainsi diverses démonstrations du même résultat et avoir une idée des performances des programmes.

- **Taille.** C'est le nombre de productions qui figurent dans la formule de démonstration plus un. Dans le cas où il n'y a que des productions à 2 antécédents, c'est le nombre d'axiomes qui y figurent. Il faut introduire le nombre de productions à cause de l'existence des productions à un antécédent. Si on prenait le nombre d'axiomes, Aa et ABACAa auraient la même taille.

Pour les démonstrations données en 2C2, on voit que l'on a :

Pour les exemples 1, 2, 3, 4, : n = 10
5, 6 : n = 12
7 : n = 17

- **Profondeur.** Les quatre premiers exemples de 2C2 ont la même taille et pourtant leur difficulté est inégale, la première étant de beaucoup la plus difficile. Un facteur important pour mesurer la difficulté d'une démonstration est la ramification du graphique qui la représente. On constate qu'en général plus l'arbre est ramifié, plus la démonstration est difficile à trouver. On voit que le premier exemple a un arbre très ramifié alors que celui du sixième est totalement linéaire : à chaque étape on fait coïncider le résultat avec un des antécédents et l'autre avec un axiome.

Pour avoir une idée de la ramification, nous allons introduire la profondeur p.

p (axiome) = 1
p (A α) = p(α)+1
p (R$\alpha\beta$) = max (p(α),p(β))+1

max (a,b) désignant le plus grand des nombres a et b.

Nous verrons plus tard en 3A que cette profondeur intervient sous le nom de niveau dans l'étude de la solution combinatoire. Par ce procédé, on ne peut espérer dépasser le niveau 6.

Pour les exemples de 2C2, on a :

1 : p = 7
2, 3, 4 : p = 9
5, 6 : p = 12
7 : p = 11

Une démonstration linéaire est caractérisée par n=p. Plus n et p seront différents, plus la démonstration sera ramifiée, donc difficile. A tailles égales, on voit que le premier exemple a le plus faible p. Une mesure de la ramification pourrait être $R = \frac{n}{p} \geq 1$. Si R = 1, on a une démonstration linéaire. Plus R est grand, plus l'arbre est ramifié.

- Nombre d'axiomes différents utilisés dans la formule de démonstration.

Les démonstrations que l'on a donné en 2C2 pour un théorème de l'axiomatique 1 n'utilisent que 2 des 3 axiomes. Plus une démonstration utilise d'axiomes différents, plus elle est difficile. Cela revient à se placer dans une axiomatique qui a plus d'axiomes, donc plus de théorèmes possibles.

Remarque - On a souvent intérêt à garder des résultats intermédiaires et à les utiliser par la suite directement. Cela revient à diminuer la taille et tout se passe comme si l'on avait davantage d'axiomes. L'un des facteurs est favorable, l'autre défavorable. On conçoit qu'il y a un compromis : il faut garder des intermédiaires, mais pas trop, car cela augmenterait dangereusement le nombre des essais possibles. Un des facteurs importants de la réussite d'un programme de démonstrations est son aptitude à choisir judicieusement les intermédiaires qu'il va garder.

En résumé, trois nombres sont utiles pour aider l'évaluation de la difficulté d'une démonstration :

- la taille
- la profondeur
- le nombre d'axiomes différents utilisé.

La difficulté croit si le premier et le dernier augmentent et si le deuxième diminue.

2D - MMTHEOREMES ET MMMTHEOREMES UTILISES PAR LE PROGRAMME

2D1 - Généralités

On va donner la liste de toutes les règles dont le programme a connaissance et qu'il va utiliser au cours de ses essais. Parmi ces règles certaines sont des MMthéorèmes et ont pour résultat de créer de nouveaux Mthéorèmes qui sont des productions. D'autres sont des MMMthéorèmes et créeront de nouveaux MMthéorèmes qui à leur tour vont permettre de démontrer des Mthéorèmes.

Il y a une autre distinction, beaucoup moins importante : certains résultats sont inclus plus ou moins explicitement dans le programme. D'autres sont lus ce qui laisse la possibilité de ne pas toujours les prendre ou d'en introduire de nouveaux. Programmer une règle a pour avantage de pouvoir utiliser des types de règles plus complexes que celles qui sont lues : ces dernières demandent la programmation d'un dispositif capable de les utiliser correctement, et qui n'a qu'une puissance limitée.

2D2 - MMthéorèmes utilisés

2D2α - *MMthéorèmes programmés*

1 - $a,b \Rightarrow c \longleftrightarrow b,a \Rightarrow c$

Ce MMthéorème est contenu de façon implicite en de nombreux points du programme. On considère comme identiques deux productions à deux antécédents qui ne différent que par l'ordre des antécédents : $p,\supset pq \Rightarrow q$ est la même production que $\supset pq,p \Rightarrow q$.

Au point de vue formule de démonstration, cela revient à permuter i et j : si $a,b \Rightarrow c$ a pour formule $\alpha i \beta j \gamma$ (resp. $\alpha j \beta i \gamma$), $b,a \Rightarrow c$ aura pour formule $\alpha j \beta i \gamma$ (resp. $\alpha i \beta j \gamma$).

2 - $a,a \Rightarrow b \longrightarrow a \Rightarrow b$

Si les deux antécédents sont identiques, on se ramène à une production à un seul antécédent.

Dans la formule de démonstration, cela va créer une particularité : il va y avoir deux occurrences de i. Si à $a,a \Rightarrow b$ correspond $\alpha i \beta j \gamma$, à $a \Rightarrow b$ correspondra $\alpha i \beta i \gamma$.

Exemple : $\supset pq$, $\supset pq \Rightarrow \supset p \wedge qq$ donnera la production $\supset pq \Rightarrow \supset p \wedge qq$

Ce MMthéorème est d'une application très rare. Si l'on utilisait beaucoup les formules de démonstration, il vaudrait mieux ne pas le garder, car cela crée des cas particuliers pour ces formules : possibilité d'avoir plusieurs occurrences de i.

3 - Si p est une variable qui ne figure pas dans a : $p \Rightarrow a \longrightarrow a$

En effet on peut substituer à p n'importe quel axiome. Cette substitution ne va pas modifier a puisque p n'y figure pas et a est bien un théorème puisque l'antécédent en est un.

Si $p \Rightarrow a$ a pour formule $\alpha i \beta$ et x est la formule d'un axiome (ou d'un théorème quelconque), $\alpha x \beta$ va être une formule de démonstration de a. On voit ici une cause d'ambiguité, puisque x peut être n'importe quelle formule de théorème.

Exemple : $p \Rightarrow \supset q \supset rq$. On peut en déduire que $\supset p \supset qp$ est un théorème.

L'utilisation de ce MMthéorème est assez fréquente. Voir par exemple la démonstration de T4 dans l'axiomatique 3.

4 - Si p est une variable ne figurant pas dans a et b, on a alors : p, a⇒b⟶a⇒b

Si αiβjγ (resp. αjβiγ) est la formule de p,a⇒b celle de a⇒b sera αxβiγ (resp. αiβxγ) où x est la formule de démonstration d'un axiome ou d'un théorème quelconque.

Exemple : p,q⇒Crq. On déduit alors p⇒Cqp.

L'axiomatique 3 utilise fréquemment ce MMthéorème.

5 - Si α = β est une définition, on peut substituer β à toute occurrence de α et inversement.

Cette règle se formalise très mal.

Exemple : Si CApqp est un théorème et que l'on ait la définition Apq=~Cp~q alors C~Cp~qp est un théorème.

Les formules de démonstration des théorèmes avant et après application de la définition sont les mêmes. L'application d'une définition n'est qu'une simple commodité d'écriture et n'apporte rien de fondamentalement nouveau.

2D2b - *MMthéorèmes lus*

1 - a;a⇒b⟶b

Application d'une production à un antécédent à un théorème. Si le théorème a pour formule α et la production βiγ, b aura pour formule βαγ.

Exemple : CCpCqrCCpqCpr+CCpqr⇒Cqr⟷CCpqCCrpCrq

Dans l'axiomatique 1 le premier a pour formule a, la production RbRRiba et le résultat : RbRRaba

2 - a⇒b;b⇒c⟶a⇒c

C'est la transivité de l'implication de la Mlangue. Si a⇒b a pour formule αiβ cela signifie que si a a pour formule i, b aura pour formule αiβ. De même pour b⇒c : si b a pour formule i, c aura pour formule γαiβδ qui est la formule de a⇒c.

Exemple : p⇒Cqp+CpCqr⇒CCpqCpr⟷Cpq⇒CCrpCrq

Le premier a, dans l'axiomatique 1, pour formule Rib, le deuxième Ria et le résultat RRiba

3 - a⇒b;b,c⇒d⟶a,c⇒d

C'est encore la transitivité de l'implication. Si a a pour formule i, b aura pour formule αiβ. Si b a pour formule i et c pour formule j, d aura pour formule γiδjθ (resp. γjδiθ). Donc si a a pour formule i et si c a pour formule j, d aura pour formule γαiβδjθ (resp. γjδαiβθ).

Exemple : p⇒Cqp+CpCqr,Cpq⇒Cpr⟷Cpq,Crp⇒Crq

Toujours dans l'axiomatique 1 les deux premières productions ayant pour formule Rib et RjRia, le résultat aura pour formule RjRRiba.

Si l'on veut modifier le deuxième antécédent, on peut permuter les antécédents et appliquer le résultat précédent.

Exemple : p⇒Cqp+Cpq,CpCqr⇒Cpr⟷p,CqCpr⇒Cqr qui aura pour formule : RRibRja.

4 - a⇒b;c,d⇒a⟶c,d⇒b

On modifie le résultat de la production à deux antécédents. Si c a pour formule i et d pour formule

j, a aura pour formule $\gamma i\delta j\theta$ (resp. $\gamma j\delta i\theta$). Mais si a a pour formule i, b aura pour formule $\alpha i\beta$. On en conclut que la formule de c,d⇒b est $\alpha\gamma i\delta j\theta\beta$ (resp. $\alpha\gamma j\delta i\theta\beta$).

Exemple : ⊃p⊃qr,⊃pq⇒⊃pr+p⇒⊃qp⟷⊃p⊃qr,⊃pq⇒⊃s⊃pr qui aura pour formule dans l'axiomatique 1 : RRjRiab.

2D3 - MMMthéorèmes utilisés

2D3a - *MMMthéorèmes programmés*

1 - a,b⇒c⇝a⟶b⇒c

Si le fait que a et b soient des théorèmes entraine que c est un théorème, alors le fait que a est un théorème entraine que si b est un théorème, c est un théorème.

On voit comme il est difficile d'exprimer ces notions dans une langue naturelle qui ne distingue pas les différents niveaux linguistiques.

Si a,b⇒c a pour formule $\alpha i\beta j\gamma$ (resp. $\alpha j\beta i\gamma$), le MMthéorème a⟶b⇒c fera correspondre au théorème a de formule δ le Mthéorème b⇒c de formule $\alpha\delta\beta i\gamma$ (resp. $\alpha i\beta\delta\gamma$) puisque a est le premier antécédent que l'on fera coïncider ultérieurement avec un théorème de formule δ.

Exemple : Si p,⊃pq⇒q a pour formule Rij, ⊃pq,p⇒q a pour formule Rji.

⊃pq,p⇒q⇝⊃pq⟶p⇒q

et ce dernier MMthéorème transformera les formules de la façon suivante : $\delta\longrightarrow Ri\delta$.

Si ⊃p⊃qp a pour formule b,p⇒⊃qp aura pour formule Rib.

2 - a,b⇒c⇝p⇒a⟶p,b⇒c où p est une variable qui ne figure ni dans a, ni dans b, ni dans c.

Si a,b⇒c a pour formule $\alpha i\beta j\gamma$ (resp. $\alpha j\beta i\gamma$), on voit que si p⇒a a pour formule $\delta i\theta$, p a pour formule i et a:$\delta i\theta$ donc p,b⇒c a pour formule $\alpha\delta i\theta\beta j\gamma$ (resp. $\alpha j\beta\delta i\theta\gamma$).

On a donc la transformation des formules

$$\delta i\theta\longrightarrow\alpha\delta i\theta\beta j\gamma$$

qui fait correspondre une production à 2 antécédents à une production à 1 antécédent.

Exemple : ⊃pq,p⇒q a pour formule Rji.

p⇒⊃qr⟶p,q⇒r transforme les formules par :

$\alpha i\beta\longrightarrow Rj\alpha i\beta$

Si on l'applique à ⊃p⊃qr⇒⊃⊃pq⊃pr de formule Ria, on a :

⊃p⊃qr,⊃pq⇒⊃pr de formule RjRia.

3 - a,b⇒c⇝c⇒p⟶a,b⇒p où p est une variable qui ne figure pas dans a, b et c.

Si a,b⇒c a pour formule $\alpha i\beta j\gamma$ (resp. $\alpha j\beta i\gamma$), on voit que si c⇒p a pour formule $\delta i\theta$, alors a,b⇒p a pour formule $\delta\alpha i\beta j\gamma\theta$ (resp. $\delta\alpha j\beta i\gamma\theta$).

Exemple : p,q⇒∧pq a pour formule Sij.

On a le MMthéorème :

∧pq⇒r⟶p,q⇒r qui à une production de formule $\alpha i\beta$ fait correspondre une production de formule $\alpha Sij\beta$.

Appliqué à la production $\wedge\supset pq\supset qr \Rightarrow \supset pr$, il donne la production $\supset pq, \supset qr \Rightarrow \supset pr$. On voit comme ce MMthéorème est important, car il établit un rapprochement entre le symbole "$\wedge$" de la langue et le "et" de la Mlangue.

2D3b - *MMMthéorèmes lus*

4 - $a \square a \longrightarrow b \Rightarrow c \rightsquigarrow b \Rightarrow c$

Il donne une manière d'utiliser les MMthéorèmes du type $a \longrightarrow b \Rightarrow c$. Si a a pour formule α et $a \longrightarrow b \Rightarrow c$ transforme la formule β en la formule $f(\beta, i), b \Rightarrow c$ aura pour formule $f(\alpha, i)$.

Exemple : $\supset p \supset qp + \supset pq \longrightarrow p \Rightarrow q \dashrightarrow p \Rightarrow \supset qp$

Si le théorème a pour formule $b, p \Rightarrow \supset qp$ aura pour formule Rib.

5 - $a \Rightarrow b \square a \Rightarrow b \longrightarrow c, d \Rightarrow e \rightsquigarrow c, d \Rightarrow e$

Si $a \Rightarrow b$ a pour formule $\alpha i \beta$ et si le MMthéorème transforme la formule $\gamma i \delta$ en une formule $f(\gamma, \delta, i, j)$, $c, d \Rightarrow e$ aura pour formule $f(\alpha, \beta, i, j)$.

Exemple : $\supset p \supset qr \Rightarrow \supset \supset pq \supset pr + p \Rightarrow \supset qr \longrightarrow p, q \Rightarrow r \dashrightarrow \supset p \supset qr, \supset pq \Rightarrow \supset pr$

Si la production a pour formule Ria et si le MMthéorème transforme une production de formule $\alpha i \beta$ en une production de formule $Rj \alpha i \beta$, on voit que le résultat aura pour formule RjRia.

Remarque - Les deux derniers MMthéorèmes ne sont que le modus ponens pour l'implication de la MMlangue : On a fixé les termes que l'on peut substituer aux variables dans l'énoncé plus général : $P \square P \longrightarrow Q \rightsquigarrow Q$.

2D4 - Conclusion

Ces règles sont les seules utilisées par le programme. On voit qu'aucune connective de la langue ne figure dans les MMthéorèmes ou les MMMthéorèmes précédents. Ceci est nécessaire puisque l'on voulait un programme très général qui puisse s'appliquer à diverses axiomatiques. On introduit les MMMthéorèmes justement pour démontrer les MMthéorèmes qui contiennent des connectives de la langue (comme $\supset pq \longrightarrow p \Rightarrow q$ ou $|p|qr \longrightarrow p \Rightarrow r$). A partir du moment où l'on a atteint ce but, on n'a plus besoins d'opérer à des niveaux métalinguistiques encore plus élevés.

Remarque 1 - Le formalisme seul permet d'exprimer un MMMthéorème, la langue naturelle confondant les connectives des différents niveaux. Il est pourtant essentiel de les distinguer et la différence apparait mieux si l'on se reporte aux formules de démonstration :

$\supset$: implication de la langue. Ne joue aucun rôle spécial, c'est une connective comme les autres.

$\Rightarrow$: implication de la Mlangue. A deux termes reliés par ce symbole correspond une formule de démonstration.

$\longrightarrow$: implication de la MMlangue. Transforme des formules de démonstration en d'autres formules de démonstration.

$\rightsquigarrow$: implication de la MMMlangue. A des formules, elle peut par exemple faire correspondre des règles qui transforment des formules en d'autres formules.

Remarque 2 - La possibilité de lire des MMthéorèmes ou des MMMthéorèmes n'a pas été tellement utilisée pour en changer d'une exploitation à l'autre. Tout au plus on a abandonné finalement le MMthéorème $a \Rightarrow b;c,d \Rightarrow a \longrightarrow c,d \Rightarrow b$ qui ne donnait en général que des résultats sans intérêt. On a aussi essayé épisodiquement d'autres MMthéorèmes comme : $a;a,b \Rightarrow c \longrightarrow b \Rightarrow c$.

Le principal avantage, beaucoup moins visible, est de faciliter la programmation. Au lieu de programmer chaque règle séparément, on a programmé un dispositif capable d'appliquer des règles : il a suffit naturellement d'un seul dispositif écrit une fois pour toutes. Il n'a malheureusement pas assez de puissance pour accepter n'importe quel type de règles et c'est pourquoi certaines ont été programmées directement. Il est par exemple difficile d'inclure dans ce dispositif la possibilité de tester qu'un antécédent se réduit à une variable qui ne figure pas ailleurs.

Remarque 3 - Le choix de ces règles vient de considérations heuristiques. On aurait très bien pu ne pas se donner la possibilité de créer de MMthéorèmes et utiliser un seul MMthéorème : $a;b;a,b \Rightarrow c \longrightarrow c$ qui est la formalisation de la manière d'utiliser une production.

Si l'on n'a pas agi ainsi, ce n'est pas pour avoir une plus grande puissance déductive : les deux procédés aboutissent au même ensemble de théorèmes. Les raisons sont purement heuristiques. La difficulté n'est pas d'appliquer des règles, mais de savoir quand les appliquer. L'expérience m'a conduit à penser que c'était plus facile quand on utilisait les règles précédentes. Tout ceci sera développé plus en détail dans le chapitre 3.

Remarque 4 - Il faudrait naturellement ajouter la règle de substitution qui est programmée et intervient très souvent dans le programme. Mais c'est imutile si l'on suppose que l'on a affaire à des schémas d'axiomes.

2E - DEUX METAMETATHEOREMES : LES THEOREMES DE DEDUCTION ET DE SUBSTITUTION

2E1 - Généralités

Nous allons étudier comment l'on pourrait donner au programme des moyens MMlinguistiques plus puissants que ceux exposés en 2D. Plus précisément, nous verrons comment on pourrait lui donner la possibilité de démontrer deux MMthéorèmes importants de la logique des propositions : les théorèmes de déduction et de substitution. Nous verrons également ce que le programme pourrait faire dans le cas où ils n'existent que sous une forme faible.

Finalement, nous indiquerons pour quelles raisons nous n'avons pas donné ces possibilités au programme.

2E2 - Le théorème de déduction

Il s'agit du MMthéorème $p \Rightarrow q \longrightarrow \supset pq$ qui est l'inverse du MMthéorème $\supset pq \longrightarrow p \Rightarrow q$ que le programme démontre aisément dans les axiomatiques de la logique des propositions qui utilisent l'implication. A eux deux, ils montrent l'équivalence de l'implication de la langue et de celle de la Mlangue : $\supset pq \longleftrightarrow p \Rightarrow q$. C'est cette équivalence qui rend difficile à saisir le concept de Mconnective.

$p \Rightarrow q \longrightarrow \supset pq$ signifie que si l'on a démontré la production $p \Rightarrow q$, alors $\supset pq$ est un théorème.

Il s'agit de démontrer ce MMthéorème : il n'existe pas dans toutes les axiomatiques. Il se peut également qu'il n'y soit que partiellement exact, c'est-à dire valable seulement pour un certain sous ensemble des productions à un antécédent.

Nous allons traiter un problème un peu plus général et chercher des conditions suffisantes pour que l'on ait le MMthéorème : $a \Rightarrow b \longrightarrow c$.

Supposons que l'axiomatique ait des productions à 2 antécédents de formule $R_k ij$ et d'énoncé $A_k, B_k \Rightarrow C_k, k=1,2,\text{--}m$ et les productions à un antécédent de formule $S_l i$ et d'énoncé $D_l \Rightarrow E_l . l=1,2,\text{--}n$. (m ou n peuvent être nuls, mais pas les deux à la fois ; dans l'axiomatique 1 : m=1;n=0). Tout théorèmes ou Mthéorème que l'on a démontré a une formule de démonstration où apparaissent les noms des axiomes, les S_l, les R_k et les i et j pour les productions.

Par hypothèse on suppose que l'on a démontré la production à un antécédent dont l'énoncé est $a \Rightarrow b$. On connait donc sa formule de démonstration qui appartient obligatoirement à l'un des 6 types suivants :

1 $R_k iT$
2 $R_k Ti$
3 $S_l i$
4 $R_k tT$
5 $R_k Tt$
6 $S_l t$

Dans ce qui précède t représente une formule contenant i donc correspondant à une production à un antécédent et T une formule ne contenant pas i donc correspondant à un théorème dont on peut déterminer l'énoncé à l'aide de l'algorithme donné en 2C1. En effet comme $a \Rightarrow b$ est une production à un antécédent, sa formule ne comporte qu'une occurrence de i et j n'y figure pas.

Pour chacun de ces cas, on va établir une condition suffisante pour que l'on puisse déduire que c est un théorème. Nous donnerons ensuite un exemple dans le cas particulier important où :

$a \Rightarrow b \longrightarrow c$ est $p \Rightarrow q \longrightarrow \supset pq$
Rij étant : $r, \supset rs \Rightarrow s$ (m=1;n=0)

Nous allons raisonner par récurrence en supposant le problème résolu pour les formules de taille inférieure à n (cf. 2C3) et nous montrerons qu'il l'est aussi pour les formules de taille n. Les trois premiers cas résoudront le cas des formules de taille 1. (Des formules de taille supérieure à 1 rentrent dans les trois premiers cas, mais toutes les formules de taille 1 appartiennent à l'un des trois premiers cas). Nous allons donc prendre les 6 cas les uns après les autres, en utilisant l'hypothèse de la récurrence pour les trois derniers.

1 - $a \Rightarrow b$ a une formule du type $R_k iT$, T étant une formule ne contenant pas i. D'après la définition des formules de démonstration, cela signifie que $A_k, B_k \Rightarrow C_k$ peut se ramener par des substitutions à : $A'_k, B'_k \Rightarrow C'_k$, que le Mthéorème $a \Rightarrow b$ peut se ramener par des substitutions à : $a' \Rightarrow b'$ et que l'on a alors $A'_k = a'$ et $C'_k = b'$.

Soit c' la transformée de c par ces substitutions : $a \Rightarrow b \longrightarrow c$ devient $a' \Rightarrow b' \longrightarrow c'$.

On voit que, pour que le théorème soit vrai dans ce cas, il suffit que l'on ait la production : $B'_k \Rightarrow c'$.

En effet s'il en est ainsi, quand on fera les substitutions qui vont permettre de démontrer le Mthéorème $a' \Rightarrow b'$, on démontrera en même temps le théorème B'_k (dont la formule de démonstration est justement T) et l'on pourra alors en déduire c'.

Bien entendu, si l'on veut avoir les résultats les plus généraux possibles, il ne faut faire que les substitutions strictement indispensables, en suivant les méthodes exposées en 2B.

En résumé on doit faire coïncider :

A_k et a
C_k et b

et il suffit alors que l'on ait la production :

$B_k^! \Rightarrow c'$

Si cette production a pour formule $\alpha i \beta$, c' aura pour formule $\alpha T \beta$. Il faut autant de productions de ce type qu'il y a de productions à 2 antécédents dans l'axiomatique, c'est à dire m.

Exemple : $A_k, B_k \Rightarrow C_k$ est $p, \supset pq \Rightarrow q$

$a \Rightarrow b \longrightarrow c$ est $r \Rightarrow s \longrightarrow \supset rs$

on fait coïncider : p et r
q et s

La production $\supset pq \Rightarrow \supset rs$ devient après substitution $\supset pq \Rightarrow \supset pq$ qui existe toujours. Il suffit de prendre pour α et β la suite vide. $\supset rs$ a pour formule T (ce qui est évident).

Remarque 1 - Il faut considérer $a' \Rightarrow b'$ et non $a \Rightarrow b$, car il se peut qu'il soit impossible d'obtenir $a \Rightarrow b$ à partir d'une formule du type $R_k i T$, mais seulement $a' \Rightarrow b'$ ou des énoncés qui s'en déduisent par substitution. Ceci sera également valable pour les cinq autres cas. Si par exemple on a RTi avec Rij: $p, \supset pq \Rightarrow q$, tout Mthéorème ayant une formule de ce type aura un énoncé du type $\supset pq \Rightarrow q$, l'énoncé du terme à substituer à p dépendant de T. En aucun cas, l'antécédent ne pourra se réduire à une variable. Il est alors inutile de demander que c soit un théorème, on peut se contenter de c' et de la production $B_k^! \Rightarrow c'$ qui est plus "faible" que $B_k^! \Rightarrow c$.

Remarque 2 - Nous ne parlons que de condition suffisante. On peut difficilement donner une condition nécessaire, car la gamme des moyens métathéoriques dont on dispose pour démontrer $a \Rightarrow b \longrightarrow c$ est vaste et mal délimitée. Quoiqu'il en soit cette condition suffisante n'est pas trop large, et l'on verra plus tard que, dans le cas de la logique des propositions, elle ne demande pas plus d'hypothèses que la démonstration classique.

2 - $a \Rightarrow b$ a une formule du type $R_k i T$, T étant une formule ne contenant pas i. Ce cas se traduit d'une façon analogue au précédent en permutant les rôles de A_k et de B_k : on fait coïncider a et B_k, b et C_k. c devient c' et il suffit d'avoir la production $A_k^! \Rightarrow c'$. Si cette production a pour formule $\alpha i \beta$, c' aura pour formule $\alpha T \beta$.

Exemple : $A_k, B_k \Rightarrow C_k$ est $p, \supset pq \Rightarrow q$

$a \Rightarrow b \longrightarrow c$ est $r \Rightarrow s \longrightarrow \supset rs$

on fait coïncider : r et $\supset pq$
s et q

Il faut que l'on ait la production $p \Rightarrow \supset rs$ qui, après substitution devient : $p \Rightarrow \supset \supset pqq$.

3 - $a \Rightarrow b$ a une formule du type $S_l i$.
Cela signifie que $a' \Rightarrow b'$, déduit de $a \Rightarrow b$ par substitution, est un cas particulier de la production $S_l : D_l \Rightarrow E_l$.

Il existe donc des substitutions telles que a et D_l d'une part, b et E_l d'autre part coïncident. Après ces substitutions c devient c'. Il y a une condition suffisante évidente : c' doit être un théorème.

Exemple : $D_l \Rightarrow E_l$ est $\supset\supset pqr \Rightarrow \supset qr$

$a \Rightarrow b \longrightarrow c$ est $s \Rightarrow t \longrightarrow \supset st$

on fait coïncider : s et $\supset\supset pqr$

t et $\supset qr$

c étant $\supset st$, c' est $\supset\supset\supset pqr \supset qr$. Il suffit donc que $\supset\supset\supset pqr \supset qr$ soit un théorème.

Remarque 1 - Il n'y a pas de formule à déterminer pour c' puisque l'on sait à priori que c est un théorème, si le théorème de déduction est exact dans l'axiomatique considérée.

Remarque 2 - Ce cas n'est pas sans intérêt, comme on pourrait le penser à première vue. Il serait absurde de demander que c soit un théorème, mais c' peut en être très différent et l'exemple précédent montre qu'il est fort possible qu'il en soit un.

Si c' n'est pas un théorème, on aura un théorème de déduction qui ne sera partiellement valable : pour les production dont la formule de démonstration ne comporte pas le groupement $S_l i$.

Il est d'ailleurs rare de rencontrer une axiomatique ayant des productions à un antécédent.

4 - $a \Rightarrow b$ a une formule du type $R_k t T$, T étant une formule ne contenant pas i, donc correspondant à un théorème, t une formule contenant i, donc correspondant à un Mthéorème. Mais sa taille est inférieure d'au moins une unité à celle de $R_k t T$ et on peut lui appliquer l'hypothèse de l'induction :

si t correspond au Mthéorème dont l'énoncé est $e \Rightarrow f$, on a $e' \Rightarrow f' \longrightarrow g'$

les lettres primées correspondant à des substitutions telles que l'on amène à coïncider :

e et a

f et b

g et c

On en déduit que g' est un théorème.

Dire que $a \Rightarrow b$ a pour formule $R_k t T$ signifie qu'il existe des substitutions telles que :

$A_k, B_k \Rightarrow C_k$ devienne $A'_k, B'_k \Rightarrow C'_k$

$e \Rightarrow f$ devienne $e'' \Rightarrow f''$ (ce n'est forcément la même substitution que celle qui donne $e' \Rightarrow f'$)

$a \Rightarrow b$ devienne $a' \Rightarrow b'$

et que : a' soit e"

A'_k soit f"

C'_k soit b'

On sait que B'_k est un théorème (il a même pour formule T). Il suffit donc que l'on ait la production : $g', B'_k \Rightarrow c'$, c' se déduisant de c par la substitution qui transforme $a \Rightarrow b$ en $a' \Rightarrow b'$.

En résumé on amène à coïncider :

$a \Rightarrow b \longrightarrow c$ et $e \Rightarrow f \longrightarrow g$ ce qui détermine g'

puis a et e

b et C_k

A_k et f

Par ces substitutions c devient c' et B_k : B'_k. Il suffit alors que l'on ait la production : $B'_k, g' \Rightarrow c'$.

Si cette production a pour formule $\alpha i \beta j \gamma$ (resp. $\alpha j \beta i \gamma$) et si, puisque l'hypothèse de la récurrence est supposée valable, à e⇒f de formule t correspond g' de formule t', à c' correspondra la formule $\alpha T \beta t' \gamma$ (resp. $\alpha t' \beta T \gamma$) si B'_k est le premier antécédent et g' le second.

Exemple : $A_k, B_k \Rightarrow C_k$ est $p, \supset pq \Rightarrow q$

$a \Rightarrow b \longrightarrow c$ est $r \Rightarrow s \longrightarrow \supset rs$

$e \Rightarrow f \longrightarrow g$ est $u \Rightarrow v \longrightarrow \supset uv$

on a donc ici g' identique à $g : \supset uv$.

On fait coïncider : r et u

s et q

p et v

et l'on doit avoir la production : $\supset pq, \supset uv \Rightarrow \supset rs$

Après substitution, on a : $\supset pq, \supset rp \Rightarrow \supset rq$.

Remarque - Il n'est pas toujours possible de faire une substitution qui fasse coïncider a⇒b et e⇒f. Dans ce cas g' n'existe pas et il y a échec : le théorème ne sera pas valable s'il y a un terme du type $R_k t T$ dans la formule de démonstration étudiée.

5 - a⇒b a une formule du type $R_k T t$, t étant une formule contenant i et T une formule ne contenant pas i. Le raisonnement de 4 reste valable en permutant les rôles de A_k et B_k.

On amène à coïncider

$a \Rightarrow b \longrightarrow c$ et $e \Rightarrow f \longrightarrow g$ ce qui détermine g'

puis a et e

b et C_k

B_k et f

Par ces substitutions c devient c' et A_k : A'_k. Il suffit alors que l'on ait la production : $A'_k, g' \Rightarrow c'$.

Si cette production a pour formule $\alpha i \beta j \gamma$ (resp. $\alpha j \beta i \gamma$), A'_k a pour formule T et à g' correspondra t' déduit de t par le procédé ainsi défini supposé valable pour les formules de taille inférieure à n. A c' correspondra la formule $\alpha T \beta t' \gamma$ (resp. $\alpha t' \beta T \gamma$).

Exemple : $A_k, B_k \Rightarrow C_k$ est $p, \supset pq \Rightarrow q$

$a \Rightarrow b \longrightarrow c$ est $r \Rightarrow s \longrightarrow \supset rs$

$e \Rightarrow f \longrightarrow g$ est $u \Rightarrow v \longrightarrow \supset uv$

D'où g' : $\supset uv$.

On fait coïncider r et u

s et q

$\supset pq$ et v

et l'on doit avoir la production : $p, \supset uv \Rightarrow \supset rs$.

Après substitution, on a : $p,\supset r\supset pq\Rightarrow\supset rq$ soit $p,\supset q\supset pr\Rightarrow\supset qr$.

b - $a\Rightarrow b$ a une formule du type $S_l t$, t étant une formule contenant i mais pour laquelle l'hypothèse de la récurrence est valable puisque sa taille est n-1.

Si t correspond au Mthéorème dont l'énoncé est $e\Rightarrow f$, on a :

$$e'\Rightarrow f'\longrightarrow g'$$

les lettres primées correspondant à des substitutions telles que l'on amène à coïncider :

e et a
f et b
g et c

On en déduit que g' est un théorème.

Dire que $a\Rightarrow b$ a pour formule $S_l t$ signifie qu'il existe des substitutions telles que :

$D_l\Rightarrow E_l$ devienne $D'_l\Rightarrow E'_l$

$e\Rightarrow f$ devienne $e''\Rightarrow f''$

$a\Rightarrow b$ devienne $a'\Rightarrow b'$

et que : a' soit e"
D'_l soit f"
E'_l soit b'

Il suffit donc que l'on ait la production $g'\Rightarrow c'$

Si cette production a pour formule $\alpha i\beta$ et si à $e\Rightarrow f$ de formule t correspond g' de formule t' (que l'on peut déterminer puisque t a une taille intérieure à n), c' aura pour formule $\alpha t'\beta$.

Exemple : $D_l\Rightarrow E_l$ est $\supset\supset pqr\Rightarrow\supset qr$
$a\Rightarrow b\longrightarrow c$ est $s\Rightarrow t\longrightarrow\supset st$
$e\Rightarrow f\longrightarrow g$ est $u\Rightarrow v\longrightarrow\supset uv$

On a donc pour g' : $\supset uv$.

On fait coïncider : s et u
$\supset\supset pqr$ et v
$\supset qr$ et t

et l'on doit avoir la production : $\supset uv\Rightarrow\supset st$.

Après substitution, on a : $\supset s\supset\supset pqr\Rightarrow\supset s\supset qr$ soit $\supset p\supset\supset qrs\Rightarrow\supset p\supset rs$

En résumé, pour que l'on ait le MMthéorème $a\Rightarrow b\longrightarrow c$, il suffit que :

- pour chaque production à deux antécédents de l'axiomatique, on ait démontré les deux productions à un antécédent et les deux productions à deux antécédents déterminées en 1, 2, 4 et 5.

- pour chaque production à un antécédent on ait démontré le théorème et la production à un antécédent déterminés en 3 et 6.

Bien entendu, cela ne s'applique que pour les productions données au départ et non à celles que l'on pourra démontrer ensuite.

On a besoin de : 2 m productions à 2 antécédents
2 m + n productions à 1 antécédent
n théorèmes.

Cela n'est pas toujours possible :

- il peut y avoir échec dans les substitutions, avec les deux causes classiques : connectives différentes ou impossibilité de faire coïncider par substitution une variable avec un terme contenant cette variable.

- le théorème ou la production nécessaire n'a pas encore été démontré, ou n'est pas un théorème ou un Mthéorème de l'axiomatique étudiée.

Même dans ce cas, il se peut que le MMthéorème soit intéressant, bien que partiellement valable : il ne s'applique que pour les résultats dont la formule ne contient pas de formules pour lesquelles le résultat manquant est indispensable. Par exemple si Rij est p,⊃pq⇒q, que l'on n'ait pas réussi à démontrer p⇒⊃⊃pqq, mais que l'on ait les trois autres productions, le théorème de déduction sera valable pour tous les Mthéorèmes dont la formule ne contient pas RTi, T étant une formule. Ce ne serait pas très grave, car il est très rare de rencontrer une formule correspondant à un Mthéorème présentant de l'intérêt où i n'apparaisse pas directement après R (cf. par exemple l'axiomatique 1).

Nous allons donner un algorithme qui détermine la formule de démonstration de c à partir de celle de a⇒b. Nous nous plaçons même dans le cas où certaines productions peuvent manquer, et il peut ainsi y avoir échec. Bien entendu, si l'on a toutes les productions et tous les théorèmes, le procédé aboutira toujours.

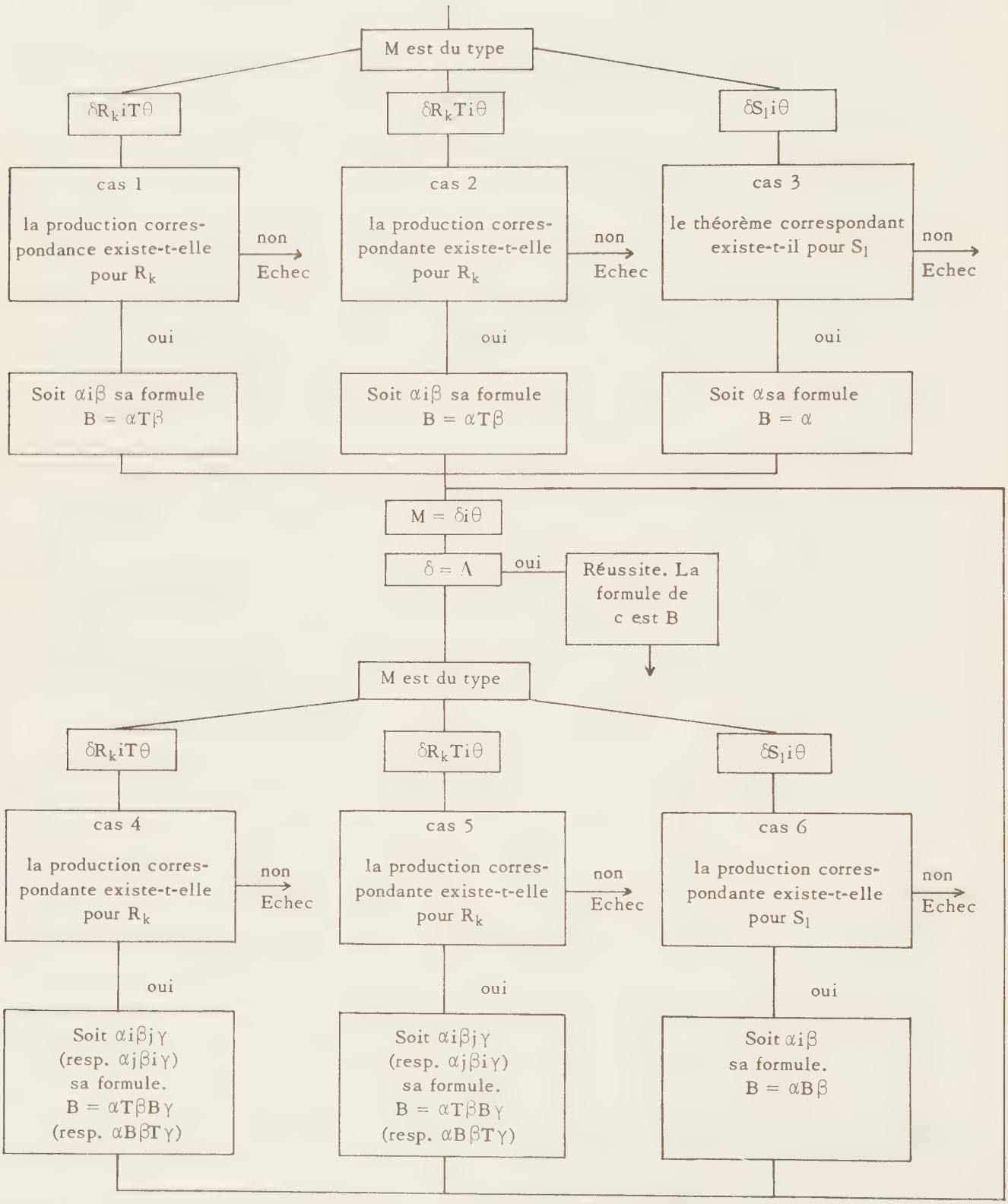
Nous ne détaillons pas la case où l'on détermine de quel type est la formule. Cela ne présente aucune difficulté : il suffit de chercher où i apparait et voir quel est le symbole qui le précède.

Les α, β, γ qui figurent dans l'organigramme sont des données. Il y en a 4 m + n. Nous ne les avons pas indicés pour ne pas compliquer l'écriture. Bien entendu ils sont différents pour des productions différentes.

Au point de vue programmation, il est intéressant de pouvoir manipuler les suites de symboles de façon commode, par exemple en utilisant un outillage de listes.

M est la formule de démonstration de a⇒b.

B sera celle de c en fin de traitement



Remarque 1 - La démonstration précédente s'étend sans difficulté au cas où l'on a des productions. à plus d'un antécédent. On peut démontrer de manière analogue d,a⇒b⟶d⇒c ou a,d⇒b⟶d⇒c. Il suffit de considérer l'antécédent d (qui correspond à i dans le premier cas et à j dans le second comme étant un axiome : on applique la méthode précédente en "figeant" la lettre qui correspond à l'antécédent d et en agissant sur j (1er cas) ou sur i (2ème cas). On aura alors une production à un antécédent qui a son tour pourra donner un théorème. Cela se généralise aisément au cas où il y a plus de 2 antécédents. On fige toutes les lettres sauf celle qui correspond à l'antécédent a.

Remarque 2 - Rapprochons cette démonstration de la démonstration classique dûe à Herbrand. Elle suppose qu'il y a une seule production à 2 antécédents : p,⊃pq⇒q et démontre :

a,b,c,...,e,f⇒g⟶a,b,c,...e⇒⊃fg

On considère d'abord les cas où :

- g est f que l'on traite avec ⊃pp
- g est un axiome ou a ou b... ou e que l'on traite avec ⊃p⊃qp.

Si l'on n'est pas dans un de ces cas, on applique la méthode d'induction et on étudie le résultat h de l'avant dernière déduction. g se déduit donc de h et de ⊃hg qui sont tous deux des résultats pour lesquels le théorème est valable. On a donc ⊃fh et ⊃f⊃hg. Il suffit alors d'appliquer ⊃⊃p⊃qr⊃⊃pq⊃pr (ou ⊃⊃pq⊃⊃p⊃qr⊃pr) pour démontrer le théorème.

On voit que l'on a besoin de :

⊃⊃p⊃qr⊃⊃pq⊃pr:a et de ⊃p⊃qp:b (en effet ⊃pp est RbRba).

Or notre méthode demande :

p⇒p qui existe toujours
p⇒⊃⊃pqq qui est RiRbRRRRbRbaaba
⊃pq,⊃rp⇒⊃rq qui est RjRRiba
p,⊃q⊃pr⇒⊃qr qui est RRibRja

Donc, dans le cas présent, on ne demande que des résultats qui peuvent se déduire des deux théorèmes que l'on utilise habituellement.

Remarque 3 - L'algorithme est valable pour toutes les formules de démonstration sauf une : i qui correspond à la production p⇒p. En toute rigueur il faudrait en plus démontrer le théorème dans ce cas particulier (dans la logique des propositions, il faut montrer que l'on a ⊃pp). Ce cas a peu d'importance, car on ne voit pas pourquoi on aurait besoin d'utiliser cette production.

Exemple 1 - On a dans l'axiomatique 1 :

⊃p⊃qr⇒⊃q⊃pr de formule RbRRRiaba

On veut démontrer ⊃⊃p⊃qr⊃q⊃pr. On a vu que dans cette axiomatique le théorème de déduction était valable. Il reste à obtenir sa formule de démonstration.

M=RbRRRiaba T=a
cas 1 - B=a

M=RbRRiba T=b
cas 4 - B=RaRRbba

M=RbRia T=a
cas 4 - B=RRaRRbbaRRaba

M=Rbi T=b

cas 5 - B=RRbbRRRaRRbbaRRabaa

(c'est la première démonstration donnée en 2C2)

Nous avons utilisé les formules des productions données dans la remarque 2.

Exemple 2 - On a : p,⊃q⊃pr⇒⊃qr de formule RRibRja

On veut avoir la formule de ⊃p⊃qr⇒⊃q⊃pr.

On va donc "figer" j et appliquer l'algorithme à i :

M=RRibRja T=b

cas 1 - B=b

M=RiRja T=Rja

cas 4 - B=RbRRRjaba

On retrouve le point de départ de l'exemple 1.

Remarque - RiRbRRRjaba est également une formule de p,⊃q⊃pr⇒⊃qr. Sa taille est de 6 au lieu de 5 pour l'exemple 1. On voit la conséquence que cela a pour l'ambiguité : dans toute démonstration utilisant le cas 5 on peut utiliser l'une quelconque des formules de démonstration. Une démonstration qui utilise n fois le cas 5 aura au moins 2^n variantes.

Exemple 3 - Supposons que l'on ait une axiomatique qui ait l'unique production :

$p,q \Rightarrow \wedge pq$ $(A_k, B_k \Rightarrow C_k)$

on veut y démontrer :

$r \Rightarrow s \longrightarrow \supset rs$ $(a \Rightarrow b \longrightarrow c)$

cas 1 - On fait coïncider : p et r
∧pq et s

et l'on veut : q⇒⊃rs soit q⇒⊃p∧pq ou p⇒⊃q∧qp

cas 2 - On fait coïncider : q et r
∧pq et s

et l'on veut : p⇒⊃rs soit p⇒⊃q∧pq

cas 4 - e⇒f⟶g s'écrit : u⇒v⟶⊃uv

On fait coïncider : r et u
∧pq et s
p et v

et l'on veut : q,⊃uv⇒⊃rs soit q,⊃rp⇒⊃r∧pq ou p,⊃qr⇒⊃q∧rp

cas 5 - On fait coïncider : r et u
∧pq et s
q et v

et l'on veut : p,⊃uv⇒⊃rs soit p,⊃rq⇒⊃r∧pq ou p,⊃qr⇒⊃q∧pr

En résumé le MMthéorème : $p \Rightarrow q \longrightarrow \supset pq$ sera exact pour cette axiomatique si l'on a les 4 productions :

$p \Rightarrow \supset q \wedge qp$
$p \Rightarrow \supset q \wedge pq$
$p, \supset qr \Rightarrow \supset q \wedge rp$
$p, \supset qr \Rightarrow \supset q \wedge pr$

Exemple 4 - Supposons que l'on ait l'unique production : $p, \supset pq \Rightarrow q \ (A_k, B_k \Rightarrow C_k)$

On veut y démontrer :

$r \Rightarrow \wedge rs \longrightarrow s \qquad (a \Rightarrow b \longrightarrow c)$

cas 1 - On fait coïncider : p et r
q et $\wedge rs$

et l'on veut : $\supset pq \Rightarrow s$ soit $\supset p \wedge ps \Rightarrow s$ ou $\supset p \wedge pq \Rightarrow q$

cas 2 - On fait coïncider : $\supset pq$ et r
q et $\wedge rs$

et l'on veut : $p \Rightarrow s$. La substitution est impossible : on doit substituer à q un terme qui le contient : $\wedge \supset pqs$. Il y a échec.

cas 4 - $e \Rightarrow f \longrightarrow g$ s'écrit $u \Rightarrow \wedge uv \longrightarrow v$

On fait coïncider : r et u
$\wedge rs$ et q
p et $\wedge uv$

et l'on veut : $\supset pq, v \Rightarrow s$ soit $\supset \wedge uv \wedge us, v \Rightarrow s$ ou $\supset \wedge pq \wedge pr, q \Rightarrow r$.

cas 5 - On fait coïncider : r et u
$\wedge rs$ et q
$\supset pq$ et $\wedge uv$

et l'on veut p, $v \Rightarrow s$. On voit qu'il y a encore échec, mais pour une autre raison que lors du cas 2 : il est impossible de faire coïncider $\supset pq$ et $\wedge uv$, les connectives étant différentes.

On voit donc que $r \Rightarrow \wedge rs \longrightarrow s$ ne sera applicable qu'à certaines formules à condition encore que l'on ait établi les Mthéorèmes : $\supset p \wedge pq \Rightarrow q$ et $\supset \wedge pq \wedge pr, q \Rightarrow r$. Il s'applique par exemple à Rib quine demande que l'utilisation du cas 1, mais si b est $\supset p \supset qp$, la formule ne correspondra à aucun énoncé : on ne peut faire coïncider $p \Rightarrow \supset qp$ et $r \Rightarrow \wedge rs$.

2E3 - Le théorème de substitution

On peut écrire le théorème de substitution lié à l'identité logique sous la forme : $\equiv pq \longrightarrow p = q$. Le signe "=" est le signe Mlinguistique introduit pour les définitions. p=q signifie qu'à toute occurrence de la suite de symboles p on peut substituer la suite q et inversement. La substitution ne se fait pas obligatoirement pour toutes les occurrences de p s'il y en a plusieurs, on peut choisir celles où on le fera. Il faut distinguer avec soin le symbole "=" de l'identité de la Mlangue "$\Leftrightarrow$". $a \Leftrightarrow b$ signifie que si a est un théorème, alors b est un théorème et que si b est un théorème, alors a est un théorème. Mais d'avoir $a \Leftrightarrow b$ ne permet pas de faire des substitutions à l'intérieur des suites de

symboles. On pourrait également démontrer ≡pq⟶p⇐⇒q qui est valable dans la logique des propositions, mais qui est beaucoup plus faible que ≡pq⟶p=q.

Supposons que nous soyons dans une axiomatique où l'on ait une connective unaire U, une connective binaire B. (On se limite à 2 connectives pour simplifier la démonstration. Elle s'étend sans difficulté à un nombre quelconque de connectives).

Nous avons, dans cette axiomatique, le MMthéorème f(p,q)⟶p⇒q f(p,q) désignant un terme où apparaissent p et q.

On veut montrer que, si l'on a les théorèmes f(P,Q) et f(Q,P), on peut substituer à toute occurrence de P:Q et à toute occurrence de Q:P. C'est à dire que si on a un théorème d'énoncé aPb, on peut en déduire que aQb est également un théorème et inversement.

Nous allons supposer que nous avons au moins une production dans chacun des trois groupes suivants :

groupe 1 : 1 - f(p,q)⇒f(Up,Uq)
2 - f(p,q)⇒f(Uq,Up)

groupe 2 : 3 - f(p,q)⇒f(Brp,Brq)
4 - f(p,q)⇒f(Brq,Brp)

groupe 3 : 5 - f(p,q)⇒f(Bpr,Bqr)
6 - f(p,q)⇒f(Bqr,Bpr)

Les productions 2, 4 et 6 seront dites "inversantes", car elles inversent l'ordre de p et de q.

Supposons par exemple que nous ayons les productions 2, 3 et 6.

Nous allons raisonner par induction sur le nombre de symboles des suites a et b. Au départ nous avons les théorèmes f(P,Q) et f(Q,P).

Supposons que l'on ait démontré f(M,N) et f(N,M), M étant de la forme cPd et N de la forme cQd, c étant une queue de a et d une tête de b (qui peuvent se réduire toutes deux à la suite vide).

aPb peut donc s'écrire : fcPdh ou fMh et aQb : fcQdh ou fNh. Trois cas sont possibles :

1 - fMh est de la forme : mUMh
2 - fMh est de la forme : mBSMh
3 - fMh est de la forme : mBMsl

S désignant un terme.

Si l'on est dans le cas 1, on applique la production 2 aux théorèmes f(M,N) et f(N,M). On voit que f(UN,UM) et f(UM,UN) sont aussi des théorèmes. On a donc un résultat analogue à l'hypothèse de départ, mais avec un symbole de plus.

Si l'on est dans le cas 2, on applique la production 3 aux théorèmes f(M,N) et f(N,M) en substituant S à r. f(BSM,BSN) et f(BSN,BSM) sont aussi des théorèmes et l'on a encore un résultat analogue à l'hypothèse de départ, mais avec au moins 2 symboles en plus.

Si l'on est dans le cas 3, on applique la production 6 et l'on voit que f(BNS,BMS) et f(BMS,BNS) sont aussi des théorèmes.

Par ce procédé, on arrivera finalement à démontrer que f(aPb,aQb) et f(aQb,qPb) sont des théorèmes.

Mais l'on a le MMthéorème f(p,q)⟶p⇒q, donc on en déduit que l'on a les Mthéorèmes : aPb⇒aQb et aQb⇒aPb. Si aPb est un théorème, aQb en est un et inversement.

Cas particulier : f(p,q) est ⊃pq. B est ⊃ U est ~ - Axiomatique 1.

On a les productions
⊃pq⇒⊃~q~p
⊃pq⇒⊃⊃rp⊃rq
⊃pq⇒⊃⊃qr⊃pr

et les théorèmes ⊃PQ et ⊃QP, on peut alors substituer P à Q et inversement.

Mais dans l'axiomatique 1 on a aussi ≡pq⇒⊃pq et ≡pq⇒⊃qp, donc le théorème de substitution y est valable, il suffit que l'on ait ≡PQ.

Nous avons démontré une forme forte du théorème de substitution. Il permet de substituer P à Q et Q à P en toutes circonstances. Il se peut qu'il ne soit que partiellement valable.

- soit qu'un des théorèmes f(P,Q) ou f(Q,P) n'existe pas. Par exemple si l'on veut substituer q à ⊃pq, c'est parfois possible comme le montrent les productions :

⊃⊃pqr⇒⊃qr
⊃s⊃⊃pqr⇒⊃s⊃qr
⊃⊃s⊃pqr⇒⊃⊃sqr

mais ce n'est pas toujours possible : on n'a pas la production ⊃pq⇒q. Cela vient de ce que ⊃p⊃qp est un théorème alors que ⊃⊃qpp n'en est pas un.

- soit que certaines productions n'existent pas. La substitution n'est alors possible que pour certains énoncés de théorèmes. Si les productions 1 et 2 n'existent pas, le théorème ne sera valable que si le terme à substituer P n'est pas inclus dans un terme de la forme UaPb. Il ne doit pas y avoir d'occurences de U qui commandent P. Il y aura échec pour BUBPqr, mais on pourra appliquer le théorème à BBPqUr.

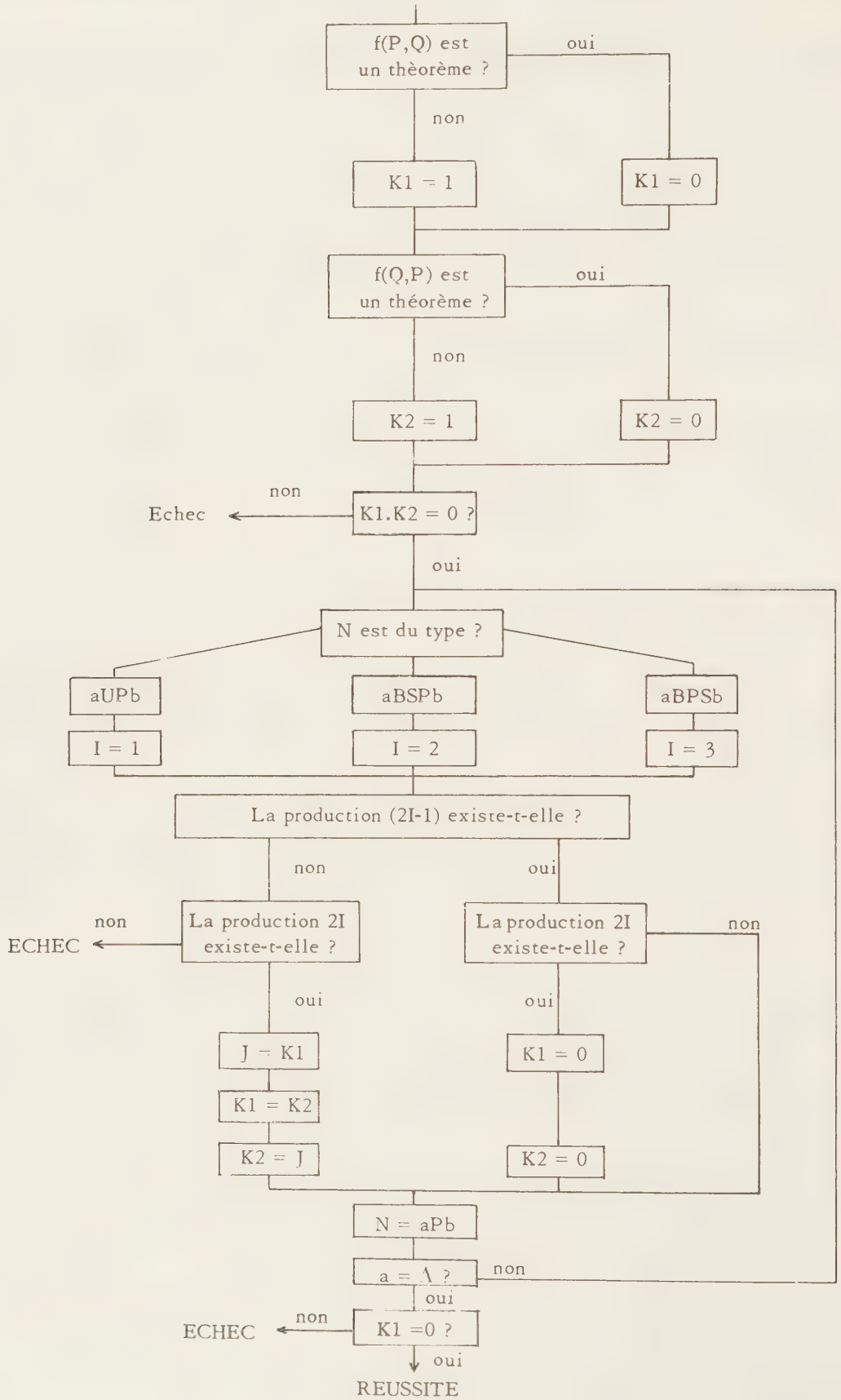
Remarque - Quand on a un seul des résultats f(P,Q) ou f(Q,P), le nombre de productions inversantes qu'il faudra appliquer indique si la substitution est possible. Si l'on a seulement f(Q,P) et que l'on veut substituer P à Q, il faut, pour que ce soit possible, que l'on ait un nombre impair de fois à appliquer une production inversante.

Donnons un organigramme qui exécute les opérations que l'on vient d'indiquer. Si ce n'est pas possible d'effectuer la substitution dans un certain cas, il sort un message d'échec.

On ne fait pas figurer le calcul des formules de démonstration qui ne pose aucune difficulté : quand on applique à un résultat intermédiaire de formule α une production de formule $\beta i \gamma$, le nouveau résultat a pour formule $\beta\alpha\gamma$.

A la fin on a f(aPb,aQb) de formule δ. Le MMthéorème f(p,q)⟶p⇒q permettra à partir de δ de déterminer la formule $\theta i \lambda$ de aPb⇒aQb et si μ est la formule de aPb, celle de aQb sera $\theta\mu\lambda$.

N est l'énoncé d'un théorème (et non pas sa formule de démonstration) mis sous la forme a Pb. On veut substituer Q à P.



Dans cet organigramme K1 et K2 sont des compteurs modulo 2 du nombre d'inversions de P et de Q. Si $K_1=0$ le théorème f(cPd,cQd) existe et si $K_2=0$ on a f(cQd,cPd). Les productions inversantes échangent les valeurs de K_1 et de K_2.

On voit les trois causes d'échec :

- on n'a ni f(P,Q), ni f(Q,P) : test $K_1.K_2=0$.
- il n'existe aucune des deux productions du groupe nécessaire.
- on a démontré seulement f(aQb,aPb) alors que l'on voulait f(aPb,aQb) : c'est le dernier test.

Exemple : f(p,q) est ⊃pq. On a le MMthéorème : ⊃pq⟶p⇒q et les productions :

2 - ⊃pq⇒⊃~q~p
3 - ⊃pq⇒⊃⊃rp⊃rq
6 - ⊃pq⇒⊃⊃qr⊃pr

Si l'on a à la fois ⊃PQ et ⊃QP il est bien évident d'après l'organigramme que la substitution de P à Q ou de Q à P sera toujours possible puisqu'au départ on a $K_1=K_2=0$ et qu'ils le resteront toujours : on ne fera qu'échanger leurs valeurs chaque fois que l'on appliquera les productions 2 ou 6.

Nous supposons que nous avons le théorème ⊃p⊃qp:f(p,⊃qp) et l'on veut substituer ⊃qp à la première occurrence de p dans ⊃~⊃rp~p.

Donc P est p et Q est ⊃qp. $K_1=0.K_2=1$.

N est ⊃~⊃rP~p.

I=2 - S est r

L'application de la production 3 donne : ⊃⊃rp⊃r⊃qp $K_1=0.K_2=1$.

N devient ⊃~P~p

I=1 - L'application de la production 2 donne : ⊃~⊃r⊃qp~⊃rp . $K_1=1.K_2=0$.

N devient ⊃P~p

I=3 - L'application de la production 3 donne : ⊃⊃~⊃rp~p⊃~⊃r⊃qp~p . $K_1=0.K_2=1$.

N devient Λ

Comme $K_1=0$ la substitution a réussi. On le voit du reste par application de ⊃pq⟶p⇒q au résultat précédent :

⊃~⊃rp~p⇒⊃~⊃r⊃qp~p

et comme ⊃~⊃rp~p est un théorème, ⊃~⊃r⊃qp~p est un théorème.

Par contre on verrait que cette substitution ne serait pas possible avec la deuxième occurrence de p, car l'on aurait pour valeur terminale de K_1:1.

Remarque 1 - S'il y a plusieurs connectives unaires ou binaires,l'organigramme subsiste à condition d'ajouter des indexations supplémentaires. Si ces connectives se ramènent par des définitions aux connectives de base, il est inutile de rien changer. Il suffit d'appliquer les définitions dans l'énoncé du théorème de façon à ce qu'il n'apparaisse plus que les connectives de base.

Remarque 2 - S'il y a plusieurs occurrences de P dans le théorème et qu'on veut faire la substitution pour ces diverses occurrences, il faut appliquer l'algorithme autant de fois qu'il y a d'occurrences. On peut même, si l'on veut, faire des substitutions différentes pour des occurrences différentes.

Remarque 3 - Nous n'avons pas donné au programme la possibilité d'utiliser la théorème de substitution,

mais nous verrons en 3B2b que l'on accorde une grande importance aux MMthéorèmes du type :

$f(p,q) \longrightarrow Brp \Rightarrow Brq$

$f(p,q) \longrightarrow Bpr \Rightarrow Bqr$

- - - - - - - - - - - -

qui lui sont liés de très près.

De même l'introduction des "parties neutres" en 3B2b attire l'attention du programme sur les productions dont l'effet se borne à substituer un terme à un autre à l'intérieur d'un théorème.

2E4 - Conclusion

Les deux MMthéorèmes ont été démontrés par récurrence, mais dans un cas on a opéré sur les symboles de la formule de démonstration et dans l'autre sur ceux de l'énoncé. S'il y a ambiguité, c'est à dire plusieurs formules pour le même énoncé, on aura dans le premier cas un comportement de l'algorithme très différent d'une formule à l'autre. Dans le deuxième on établira dans tous les cas le Mthéorème $aPb \Rightarrow aQb$ de formule $\alpha i \beta$ et c'est à ce moment seulement que l'ambiguité apparaitra : si deux formules de démonstration du théorème où P figure sont γ et δ, celui où Q figure aura pour formules $\alpha\gamma\beta$ et $\alpha\delta\beta$. Il y aura toujours ambiguité, mais l'algorithme aura traité toutes les variantes de la même façon.

Il aurait été possible de donner au programme la possibilité de démontrer et d'utiliser ces deux MMthéorèmes. Il aurait suffit pour cela de lui donner davantage de possibilités au niveau de la MMMlangue. On ne l'a pas fait pour diverses raisons.

1 - On a essayé de généraliser ces théorèmes, mais ils restent quand même très liés à des à des axiomatiques particulières. Notre but n'étant pas de trouver des théorèmes, mais de trouver comment on peut en trouver, nous avons préféré sacrifier les performances à la généralité. Si on avait limité le programme à une axiomatique particulière, on aurait pu avoir d'excellentes performances en profitant des particularités de cette axiomatique. Il aurait alors mieux valu parler d'intelligence du programmeur plutôt que d'intelligence artificielle.

2 - Pour utiliser le théorème de déduction sous une forme faible, il faut garder en mémoire les formules de démonstration des théorèmes. Cela ferait perdre de la place :

- d'abord par les instructions nécessaires pour établir les formules de démonstration des résultats nouveaux.
- ensuite par le stockage de ces formules volumineuses, même si l'on introduit des définitions.

Le programme étant déjà à court de mêmoire, cela aurait des conséquences très gênantes.

3 - Pour que l'utilisation de ces MMthéorèmes soit vraiment intéressante, il faudrait que le programme ait la possibilité de les manipuler. Cela obligerait à lui fournir au niveau de la MMMlangue des moyens qu'il n'a pas.

Prenons pour exemple le cas du théorème de déduction sous sa forme classique : $p \Rightarrow q \longrightarrow \supset pq$

Supposons que l'on ait le théorème $\supset\supset p \supset qr \supset\supset pq \supset pr$. On voit que l'on peut déduire successivement :

$\supset p \supset qr \Rightarrow \supset\supset pq \supset pr$	par $\supset pq \longrightarrow p \Rightarrow q$
$\supset p \supset qr, \supset pq \Rightarrow \supset pr$	par $p \Rightarrow \supset qr \longrightarrow p, q \Rightarrow r$
$\supset p \supset qr; \supset pq \longrightarrow \supset pr$	par $a, b \Rightarrow \rightsquigarrow a; b \longrightarrow c$
$\supset p \supset qr; \supset pq \longrightarrow p \Rightarrow r$	par $\supset pq \longrightarrow p \Rightarrow q$
	et $a \longrightarrow b \Box b \longrightarrow c \rightsquigarrow a \longrightarrow c$

$p \Rightarrow \supset qr \longrightarrow \supset p \supset qr$
$p \Rightarrow q \longrightarrow \supset pq$
} par le théorème de déduction

$p \Rightarrow \supset qr ; p \Rightarrow q \longrightarrow q \Rightarrow r$ par $a \longrightarrow b \Box b ; c \longrightarrow d \rightsquigarrow a, c \Rightarrow d$
et $a \longrightarrow b \Box c ; b \longrightarrow d \rightsquigarrow c, a \Rightarrow d$

Pour établir cela nous avons été obligés de faire intervenir 4 MMMthéorèmes nouveaux. Il aurait fallu les donner au programme et lui apprendre à les manipuler à bon escient ce qui n'a rien d'évident.

Par contre cela aurait très certainement amélioré ses performances en lui donnant une vue plus profonde de l'intérêt des théorèmes. On vient de montrer le lien qui existe entre $\supset\supset p \supset qr \supset\supset pq \supset pr$ et $p \Rightarrow \supset qr ; p \Rightarrow q \longrightarrow p \Rightarrow r$ qui est lié de près à la production $\supset qr, q \Rightarrow r$. On aurait vu de même le lien de $\supset\supset pq \supset\supset qr \supset pr$ avec $p \Rightarrow q ; q \Rightarrow r \longrightarrow p \Rightarrow r$. On aurait pu faire ainsi apparaitre de façon beaucoup plus naturelle l'intérêt de certains résultats ; alors que dáns le programme on a dû prendre un artifice : on a considéré arbitrairement des types de MMthéorèmes privilégiés, comme le type substitutif.

4 - Il faudrait aussi que le programme puisse choisir le type de théorèmes de substitution ou de déduction qu'il essayerait de démontrer.

Les logiciens ont été amenés à considérer $p \Rightarrow q \longrightarrow \supset pq$ parce qu'ils avaient justement créé la logique pour qu'il y ait une certaine affinité entre "$\supset$" et "$\Rightarrow$". C'est un renseignement qu'il faut donner au programme si l'on veut qu'il puisse avoir l'idée de démontrer ce MMthéorème particulier.

En résumé, nous n'avons pas donné au programme la possibilité d'utiliser les théorèmes de déduction et de substitution. Mais nous avons montré qu'il était fort possible d'écrire un programme qui puisse les trouver. Il faudrait lui donner des moyens MMMlinguistiques très puissants, non seulement pour la démonstration, mais surtout pour qu'il puisse utiliser pleinement ce qu'il a trouvé.

CHAPITRE III

ASPECT HEURISTIQUE

3A - IMPOSSIBILITE DE L'APPROCHE COMBINATOIRE

Il est très facile d'énumérer tous les théorèmes d'un système formel. Prenons par exemple le cas d'un système formel ayant une seule production à 2 antécédents. On peut construire un graphe qui donne tous les théorèmes possibles : le premier niveau comprend les axiomes. Le deuxième niveau est formé des résultats de la production obtenus en faisant coïncider par des substitutions les axiomes avec les antécédents et ceci en utilisant toutes les combinaisons possibles... Le n ième niveau est formé des résultats de la production obtenus en faisant coïncider par des substitutions les éléments des niveaux précédents avec les antécédents en utilisant toutes les combinaisons possibles avec toutefois la contrainte : au moins un des deux résultats mis en jeu lors d'un essai appartienne au niveau (n-1) (Ceci pour éviter de faire plusieurs fois le même essai).

On se rend compte que la croissance du nombre de résultats va être très rapide. Elle va être limitée par les échecs d'une manière difficilement évaluable et qui varie d'une axiomatique à l'autre. Toujours dans le cas d'une production à deux antécédents et en supposant qu'il n'y a pas d'échecs, le nombre de résultats P_n démontrés au niveau n est donné par :

$$P_n = P_{n-1} \times \left[P_{n-1} + 2(P_{n-2} + P_{n-3} + \ldots + P_1)\right]$$

où P_1 est le nombre d'axiomes.

Cela donne une croissance du type $K^{(2^n)}$ beaucoup plus rapide que l'exponentielle. Ceci est une borne supérieure limitée par les échecs et aussi par l'ambiguité : la méthode précédente énumère en réalité les démonstrations. S'il y a ambiguité, plusieurs démonstrations aboutissent au même résultat et il suffit d'en garder un exemplaire. Quoiqu'il en soit, même avec ces corrections, la croissance est extrêmement rapide.

Le problème est beaucoup plus grave que pour le jeu d'échecs où le nombre de possibilités à un niveau est inférieur à celui du niveau précédent multiplié par une constante K (on a d'ailleurs $K \ll 333$, une valeur plus précise étant difficile à évaluer). Or on sait bien que même dans ce cas plus favorable, il est impossible pratiquement d'utiliser la méthode qui consiste à examiner tous les cas possibles.

A titre indicatif donnons quelques valeurs de P_n pour diverses valeurs de P_1 (nombre d'axiomes)

P_1	1	2	3	4	5
P_2	1	4	9	16	25
P_3	3	32	135	384	875
P_4	21	1408	20465	162816	818125
P_5	651	2.10^6	4.10^8	$2,5.10^{10}$	6.10^{11}
P_6	457653	4.10^{12}	$1,6.10^{17}$	6.10^{20}	3.10^{23}
P_7	2.10^{11}	$1,6.10^{25}$	$2,5.10^{34}$	3.10^{41}	10^{47}
P_8	4.10^{22}	$2,5.10^{50}$	6.10^{68}	10^{83}	10^{94}

Même dans le cas d'un seul axiome il est illusoire par ce procédé d'espérer dépasser le niveau 6 sur les calculatrices les plus puissantes qui existeront dans un proche avenir. Or la plupart des démonstrations de résultats importants dépassent ce niveau.

Le niveau que l'on vient de définir coïncide avec la profondeur définie en 2C3. Or il y avait des démonstrations de profondeur 12 et, pour des commodités d'exposition, on n'a donné que des démonstrations relativement courtes.

Si l'on utilisait quand même ce procédé, il faudrait de toutes les façons un crible qui indique les résultats intéressants, car la plupart des théorèmes n'ont pas le moindre intérêt. A titre d'exemple voici quelques théorèmes de l'axiomatique 1 :

CpCqCrCsq ; CpCqCrCsCtq ; CpCqCrCsCtCuq ; ...

CNNpCpq ; CNNNNNpCpq ; CNNNNNNNpCpq ; ...

CNNNNpp ; CNNNNNNpp ; ...

CCNpCNqpCNCNpq ; CCCCNpNqqCNpNqCCCNpNqqp ; ...

Il n'y a pas d'autre solution que de tailler très sévèrement ce graphe pour avoir une chance de trouver en un temps raisonnable sur une machine ayant une mémoire limitée des théorèmes intéressants. On a pour cela deux moyens :

- éliminer des résultats. On ne considérera pas les combinaisons où le résultat intervient comme un des antécédents.

- éliminer des essais. On ne fera pas toutes les combinaisons possibles des résultats que l'on juge dignes d'intérêt.

Un autre moyen d'action d'une nature très différente ne va pas agir sur la taille du graphe, mais sur l'ordre dans lequel on le décrit : à chaque instant le programme va faire l'essai qu'il estime avoir le plus de chance de mener à un résultat important. Théoriquement cela n'apporte rien de plus, mais pratiquement on a un temps machine et une mémoire limités et on veut avoir le plus tôt possible ce qui nous intéresse.

Nous allons voir plus en détail comment l'on a réalisé ces trois modes d'action :

- élimination de certains résultats.
- ne pas faire toutes les combinaisons de résultats.
- essayer d'abord les combinaisons qui promettent le plus.

3B - ELIMINATION DE RESULTATS

3B1 - Etude de la force d'un résultat

a - Introduction

Le mathématicien sent bien que certains de ses résultats sont plus profonds ou plus forts que d'autres, mais il lui est difficile de préciser clairement ce qu'il entend par profondeur ou par force. On va essayer de préciser cette intuition. Il semble qu'il y ait un mélange de trois notions différentes liées à :

- la forme du résultat. Un résultat compact, qui s'exprime à l'aide de peu de symboles semble en général plus profond qu'un résultat qui demande plusieurs lignes pour le décrire.

- la démonstration du résultat. Si on doit pour l'obtenir passer par des étapes peu intuitives, sa démonstration sera difficile à trouver et le résultat aura l'air plus profond. On ne parle pas ici de la longueur de la démonstration : une démonstration très longue peut être facile si chaque pas appelle automatiquement le suivant. C'est ce qui se passe souvent en géométrie analytique. Inversement une démonstration très courte peut être très difficile à découvrir. Une démonstration est difficile si elle va contre les heuristiques les plus naturelles et si son arbre est très ramifié.

- les résultats qui peuvent s'en déduire. Un résultat est fort si il en résulte facilement un grand nombre de résultats intéressants.

A cela il faudrait ajouter des considérations sémantiques liées à l'interprétation que l'on donne au résultat. Nous ne pouvons les introduire ici, car nous avons décidé de rester dans le domaine syntaxique.

Nous allons par la suite proposer trois définitions qui mélangent plus ou moins heureusement les notions précédentes. Nous n'emploierons que le mot force.

Il y a un cas où la notion de force apparait évidente. Si a peut se déduire de b par des substitutions, nous écrirons b>a(ou a<b)

Exemples : $Cpp>CCpqCpq$ p/Cpq

$CpCqp>CpCpp$ $p/p;q/p$

$CpCqp>CpC\sim qp$ $p/p;q/\sim q$

Malheureusement il est très rare que des théorèmes soient ainsi liés. Nous allons essayer de l'étendre de diverses façons, mais dans tous les cas, nous dirons que si a se déduit de b par substitution : a<b.

Pour étendre cette notion nous allons classer les productions en 4 classes. Soit la production $c \Rightarrow d$ et supposons que l'on ait : $a+c \Rightarrow d \dashrightarrow b$.

Si la production appartient à la classe forte, nous dirons que a<b.

Si la production appartient à la classe faible, nous dirons que a>b.

Si la production appartient à la classe égale, nous dirons que a=b.

Si la production appartient à la classe spéciale, nous dirons que a et b ne sont pas comparables.

Nous avons ramené le problème de déterminer la force relative de théorèmes à celui de classer des productions. Il est bien entendu que l'on a fait dans le théorème toutes les substitutions nécessaires pour qu'on n'ait plus à en faire au moment d'appliquer la production. Si l'on veut appliquer $CpCqr \Rightarrow CqCpr$ à Cpp, on considérera d'abord le résultat plus faible $CCpqCpq$ et c'est à celui-ci que l'on appliquera la production.

Graphiquement, nous pourrons représenter les résultats d'une axiomatique en les liant par une flèche horizontale allant de a vers b s'il y a une production égale qui permet de passer de a à b, montante si c'est une production forte, descendante si elle est faible. Si on y passe par une substitution, on mettra une flèche verticale dirigée vers le bas.

Remarque 1 - Il se peut que l'on trouve le même énoncé à différents niveaux. Cela est dû à ce que nos définitions de la force, en faisant intervenir la démonstration, vont être liées aux démonstrations : s'il y a ambiguité, il n'y aura aucune raison pour que les diverses démonstrations du même énoncé présentent la même difficulté.

Remarque 2 - Tout ceci n'a qu'une valeur heuristique. On ne doit pas exiger dans ce domaine la rigueur que l'on demande pour effectuer une dérivation. Une méthode heuristique n'est pas tenue de toujours donner de bons résultats. Il suffit qu'en général elle ait de bons résultats. C'est dans cet esprit qu'il faut admettre l'existence de productions spéciales qui ne permettent pas d'évaluer la force relative des théorèmes. Il suffit qu'il n'y en ait pas trop.

Remarque 3 - On peut considérer que l'application de la définition a=b correspond au point de vue force à l'application de la production a⇒b ou b⇒a selon le sens dans lequel on utilise la définition.

b - Définition de la force basée sur l'idée de substitution

Une production est dite égale si, par substitution, l'antécédent peut devenir le résultat et inversement.

Exemple : p⇒p
Cpqr⇒CqCpr
Cp~q⇒Cq~p
Apq⇒Aqp

Une production sera dite forte si l'antécédent peut se déduire du résultat par substitution et si elle n'est pas égale.

Exemple : CCpqr⇒Cqr
C~p~q⇒Cqp
Apq⇒p
Vpp⇒Vpq

Une production sera dite faible si le résultat peut se déduire de l'antécédent par substitution et si elle n'est pas égale.

Exemple : p⇒Cqp
Cpq⇒CCrpCrq
CpCqr⇒CCpqCpr
Apq⇒App
Cpq⇒C~q~p

Une production qui ne rentre dans aucune des catégories précédentes est dite spéciale.

Exemple : Epq⇒Cpq
CCpqr⇒C~pr
CCpqr⇒CsCqr

Dans la plupart des axiomatiques, cette dernière catégorie comporte assez peu de productions intéressantes.

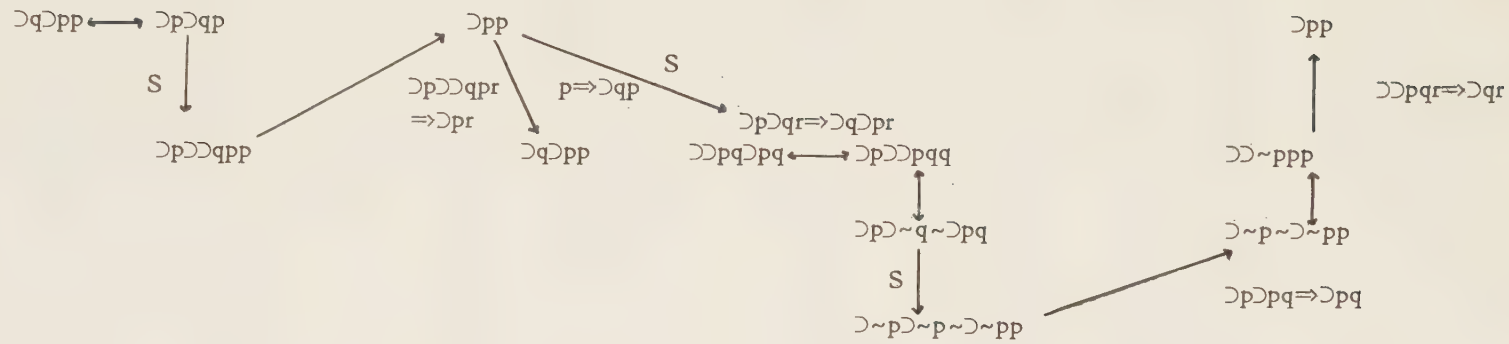
Cette définition a pour intérêt d'être très proche de la définition intuitive où un théorème qui se déduit d'un autre par substitution est dit plus faible. Il arrive même que a s'obtienne à partir de b soit par substitution, soit par action d'une production faible.

Exemple : Cpp>C~p~p. On peut en effet faire dans le premier la substitution p/~p. Mais on peut aussi lui appliquer la production faible : Cpq⇒C~q~p.

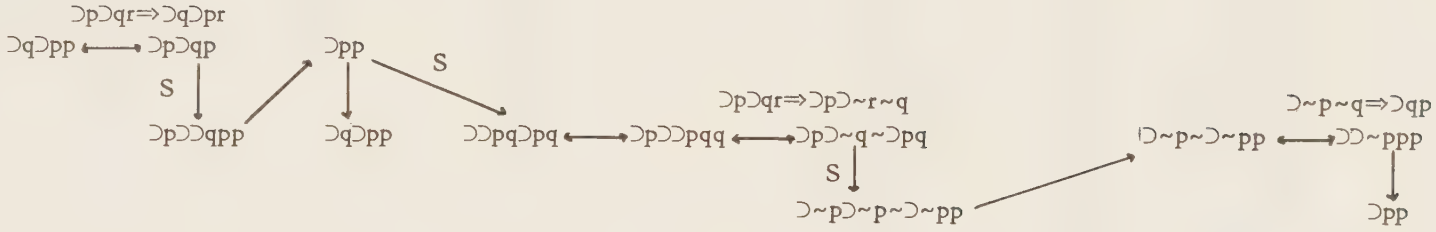
Graphe par force - S indique une substitution

Deux premières définitions

(il se trouve
que les graphes
sont les mêmes)



Troisième définition



Classons quelques théorèmes par ce procédé :

$Cpp>CCpqCpq$, car on peut faire dans le premier la substitution p/Cpq.

$CCpqCpq=CpCCpqq$, car on passe de l'un à l'autre par application de la production égale : $CpCqr\Rightarrow CqCpr$.

Donc $Cpp>CpCCpqq$

Remarque 1 - Nous donnons un exemple où nous classerons des théorèmes de l'axiomatique 1 à l'aide des 3 définitions de la force que nous envisageons. Les démonstrations les plus difficiles sont celles où il faut d'abord descendre pour remonter, car on est obligé de passer par une étape peu intéressante. On voit qu'il ne doit pas être facile de démontrer $CC\sim ppp$. Remarquons que les deux démonstrations de $CqCpp$ le mettent à des niveaux différents.

Remarque 2 - Les définitions correspondent en général à des productions spéciales, car en général les connectives des deux membres sont différentes.

c - *Définition de la force basée sur le nombre de connectives*

Soient u_1, b_1 et u_2, b_2 les nombres de connectives unaires et binaires de l'antécédent et du résultat.

- une production est égale si $u_1=u_2$ et $b_1=b_2$.

Exemple : $CpCqr\Rightarrow CqCpr$
$Vpq\Rightarrow Vpp$
$Cp\sim q\Rightarrow Cq\sim p$
$\equiv pq\Rightarrow Cpq$
$CCpqr\Rightarrow CsCqr$

- une production est forte si elle n'est pas égale et si $b_1>b_2$ et $u_1>u_2$

Exemple : $CCpqr\Rightarrow Cqr$
$C\sim p\sim q\Rightarrow Cqp$
$\wedge pq\Rightarrow p$

- une production est faible si elle n'est pas égale et si $b_1<b_2$ et $u_1<u_2$

Exemple : $p\Rightarrow Cqp$
$Cpq\Rightarrow CCrpCrq$
$Cpq\Rightarrow C\sim q\sim p$

- une production qui n'appartient à aucune des classes précédentes est spéciale.

Exemple : $C\sim\sim pq\Rightarrow CpCrq$
$CCpqr\Rightarrow C\sim pr$

Cette définition est assez grossière. Elle insiste uniquement sur la compacité du résultat.

Remarque 1 - Il y a un lien assez étroit avec la première définition. Toute production égale au sens précédent le reste.

Une production forte au sens précédent le reste ou devient égale ($Vpp \Rightarrow Vpq$).

Une production faible au sens précédent le reste ou devient égale ($\wedge pq \Rightarrow \wedge pp$).

Une production spéciale au sens précédent le reste ou devient faible ($\equiv pq \Rightarrow \supset r \supset qp$) ou forte ($\wedge \equiv pqr \Rightarrow \equiv pq$) ou égale ($\equiv pq \Rightarrow \supset pq$).

On a donc certainement une classe spéciale plus réduite.

Remarque 2 - Une définition peut être classée: $\sim \supset p \sim q = \wedge pq$ est forte si on remplace le premier membre par le second et faible dans l'autre sens. Une définition peut être spéciale : $|pp = \sim p$. Une définition pourrait être égale, mais on n'en verrait guère l'intérêt, car en général elles sont introduites pour diminuer la taille des énoncés.

Remarque 3 - On pourrait faire disparaitre la classe spéciale en comparant non pas les u et b séparément, mais les valeurs d'une fonction de u et de b.

Si $f(u_1,b_1) < f(u_2,b_2)$ la production serait faible.

Remarque 4 - Une substitution qui n'ajoute pas de connectives et ne fait que changer des noms de variables laisse les résultats au même niveau : par exemple $\supset p \supset qp$ et $\supset p \supset pp$. Cela est assez peu satisfaisant. Il n'empêche qu'il y a des cas où cette définition de la force est utile.

d - Définition de la force basée sur la réciprocité

Dans ce qui suit nous supposerons que les productions sont données sous leur forme la plus condensée ; cela veut dire que l'on ne peut pas les obtenir en faisant des substitutions dans d'autres productions.

Une production $a \Rightarrow b$ est faible si $b \Rightarrow a$ n'est pas une production.

Exemple : $\wedge pq \Rightarrow p$
$p \Rightarrow Vpq$
$\supset pq \Rightarrow \supset \supset rp \supset rq$
$\supset \supset pqr \Rightarrow \supset qr$
$p \Rightarrow \supset qp$

Une production $a \Rightarrow b$ est forte si $b \Rightarrow a$ est une production que l'on peut dériver par des substitutions d'une autre production.

Exemple : $p \Rightarrow \wedge pp$ car il y a $\wedge pq \Rightarrow p$ (ou $\wedge pq \Rightarrow q$)
$Vpp \Rightarrow p$ car il y a $p \Rightarrow Vpq$
$\supset p \supset pq \Rightarrow \supset pq$ car on a : $p \Rightarrow \supset qp$
$\supset p \supset qr \Rightarrow \supset \supset pq \supset pr$ car on a : $\supset \supset pq \supset rs \Rightarrow \supset r \supset qs$
$\supset p \supset qr \Rightarrow \supset \supset qp \supset qr$ car on a : $\supset \supset pqr \Rightarrow \supset qr$
$\supset p \supset \supset qpr \Rightarrow \supset pr$ car on a : $\supset pq \Rightarrow \supset p \supset rq$

Une production qui n'est ni forte ni faible est égale. Cela signifie que $b \Rightarrow a$ est une production que l'on ne peut pas obtenir par des substitutions dans une autre production.

Exemple : $\supset pq \Rightarrow \supset \sim q \sim p$

$\supset p \supset \sim q \sim r \Rightarrow \supset p \supset rq$

Si a⇒b est une production égale, b⇒a est bien évidemment une production égale. Un cas particulier très fréquent est celui des productions "autoréciproques" où b⇒a est la même production que a⇒b.

Exemple : $\supset p \supset qr \Rightarrow \supset q \supset pr$
$\supset p \sim q \Rightarrow \supset q \sim p$
$\supset p \supset \sim qr \Rightarrow \supset p \supset \sim rq$
$\wedge pq \Rightarrow \wedge qp$

Cette définition est très intéressante pour plusieurs raisons :

- il n'y a pas de classe spéciale : toute production est soit forte, soit faible, soit égale.

- elle correspond bien à une idée intuitive de la force des théorèmes : le théorème B est plus fort que le théorème A si, ayant A on peut en déduire B, mais ayant B on peut en déduire un résultat dont A n'est qu'un cas particulier obtenu par une certaine substitution. Si par exemple on a un théorème du type $\supset P \supset PQ$, on aura par $\supset p \supset pq \Rightarrow \supset pq$ que $\supset PQ$ est un théorème. Mais si l'on a $\supset PQ$ on peut en déduire par $p \Rightarrow \supset qp$ que $\supset R \supset PQ$ est un théorème (bien plus général que $\supset P \supset PQ$).

Mais on peut lui faire de graves critiques :

- La classification change d'une axiomatique à l'autre (les exemples sont donnés dans l'axiomatique 1) et évolue même au fur et à mesure que l'on découvre de nouveaux résultats. Au départ on considère $\supset \sim p \sim q \Rightarrow \supset qp$ comme faible et ceci jusqu'au moment où l'on aura démontré $\supset pq \Rightarrow \supset \sim q \sim p$. Cette définition, très intéressante pour une étude à postériori, se prête mal à une estimation a priori de l'intérêt.

- On peut avoir l'impression que la relation > n'est pas transitive : on peut avoir a>b, b>c et a<c. Ceci était évité avec les définitions précédentes, car les productions fortes dans les deux sens que l'on avait défini avaient toujours pour conséquence de simplifier les résultats. Ici ce n'est pas le cas et une production forte comme $\supset p \supset qr \Rightarrow \supset \supset pq \supset pr$ peut très bien les rendre plus complexes.

Exemple :
$\supset \sim \sim pp$
S ↓ $\quad \supset \sim p \sim q \Rightarrow \supset qp$
$\supset \sim \sim \sim p \sim p \longleftrightarrow \supset p \sim \sim p$
S ↓ $\quad \supset \sim p \sim q \Rightarrow \supset qp$
$\supset \sim p \sim \sim \sim p \longleftrightarrow \supset \sim \sim pp$

on voit que cela conduit à poser $\supset \sim \sim pp > \supset \sim \sim pp$

Mais ce paradoxe n'est qu'apparent et est dû à l'ambiguité. Cela vient de ce que l'on raisonne sur les énoncés. Chacun des deux énoncés : $\supset \sim \sim pp$ a une démonstration différente, la deuxième étant plus complexe que la première, car on lui a appliqué deux fois la production $\supset \sim p \sim q \Rightarrow \supset qp$.

A partir du moment où l'on applique la relation > aux formules de démonstration et non aux énoncés, la paradoxe disparait.

Les productions fortes sont souvent très importantes, car il leur arrive de jouer le même rôle que la production inverse (qui est faible).

Exemple : $\supset \supset pq \supset qr \Rightarrow \supset qr$ est forte car $\supset qr \Rightarrow \supset \supset pq \supset qr$ se déduit par substitution de $p \Rightarrow \supset qp$.

$\supset p \supset qq + \supset \supset pq \supset qr \Rightarrow \supset qr \dashrightarrow \supset pp$

Si la production $\supset qp \Rightarrow p$ existait, on aurait

$$\supset p \supset qq + \supset qp \Rightarrow p \dashrightarrow \supset pp$$

c'est à dire exactement la même résultat.

Les théorèmes situés au sommet du graphe par force sont souvent pris comme axiomes par les logiciens.

Remarquons que les définitions étant essentiellement réversibles jouent un rôle analogue aux productions égales.

Remarque - On aurait pu envisager une autre définition de la force analogue à celle-ci, mais basée sur l'ambiguité :

une production A est plus forte que la production B si elles donnent le même résultat quand B est appliqué au théorème T et A à un théorème se déduisant de T par substitution.

Exemple : 1 T est $\supset \sim p \supset pq$

A est $\supset \sim \sim pq \Rightarrow \supset pq$

B est $\supset p \supset qr \Rightarrow \supset q \supset pr$

car $\supset \sim \sim p \supset \sim pq + \supset \sim \sim pq \Rightarrow \supset pq \dashrightarrow \supset p \supset \sim pq$

$\supset \sim p \supset pq + \supset p \supset qr \Rightarrow \supset q \supset pr \dashrightarrow \supset p \supset \sim pq$

2 T est $\supset \supset \sim p \sim q \supset qp$

A est $\supset \supset pq \supset \sim r \sim s \Rightarrow \supset q \supset sr$

B est $\supset \supset pqr \Rightarrow \supset \sim pr$

car $\supset \supset \sim \sim p \sim \sim q \supset \sim q \sim p + \supset \supset pq \supset \sim r \sim s \Rightarrow \supset q \supset sr \dashrightarrow \supset \sim \sim p \supset qp$

$\supset \supset \sim p \sim q \supset qp + \supset \supset pqr \Rightarrow \supset \sim pr \dashrightarrow \supset \sim \sim p \supset qp$

Malheureusement il faudrait démontrer que cette définition est cohérente et on n'a pas assez d'exemples de tels cas pour bâtir une échelle de force des productions.

e - Conclusion

Aucune de ces définitions n'est vraiment satisfaisante. Il est difficile de formaliser un concept. intuitif qui est très flou : le peu de précision qui règne dans notre cerveau lui permet d'avoir des contradictions qui choquent dès que l'on essaye de mettre au net ce qui s'y passe. La définition la plus satisfaisante parmi les trois étudiées est sans doute la troisième. Malheureusement elle évolue avec les connaissances que l'on accumule. Cela la rend surtout utilisable pour aider la démonstration de quelques conjectures difficiles quand l'étude de l'axiomatique est déjà très avancée.

Il est intéressant d'appliquer les productions fortes et l'on a utilisé les définitions précédentes pour classer les productions en deux groupes. Grosso modo les fortes appartiennent au premier, les faibles au second et les égales aux deux. Celles du premier groupe seront par la suite utilisées pour démontrer des théorèmes nouveaux, celles du second dans les essais de démonstrations de conjectures où l'on travaille "à l'envers" : on part de la conjecture et l'on en déduit d'autres conjectures que l'on a intérêt à avoir plus faibles, donc plus faciles à prouver.

La définition de la force que l'on a programmé pour faire cette classification est un compromis entre les deux premières définitions. La troisième définition n'a été utilisée que pour reconnaitre

si une production est du groupe "égale" (cela pour éviter d'appliquer successivement a⇒b puis b⇒a) Il est probable que d'en tenir compte aurait amélioré les performances en fin d'étude des axiomatiques où le programme actuel s'essouffle quelque peu.

Remarque - Dans tout ce qui suivra nous utiliserons le signe > uniquement pour indiquer qu'un résultat peut se déduire d'un autre par des substitutions (qui ne se réduisent pas à une simple permutations des noms de variables).

Exemples : $Cp Cqp > Cp C\sim qp$

$Cp Cqp > Cp Cpp$

$p \Rightarrow Cqp > p \Rightarrow C\sim qp$

Dans $CCpqr \Rightarrow Cqr$ on a $CCpqr < Cqr$

Dans $p \Rightarrow Cqp$ on a $p > Cqp$

$<$ désignera la relation inverse : si $\alpha > \beta$ alors $\beta < \alpha$

$=$ désignera l'identité de deux suites de symboles à une permutation des noms de variable près :

$Cp Cqp = Cq Cpq$

$C\wedge pqp = C\wedge rqr$

$\alpha \geqslant \beta$ est une abréviation pour $\alpha > \beta$ ou $\alpha = \beta$

$\alpha \leqslant \beta$ est une abréviation pour $\alpha < \beta$ ou $\alpha = \beta$

3B2 - Détermination de l'intérêt d'un résultat

a - Introduction

L'intérêt que l'on va définir ici va être utilisé au cours des essais du programme pour tailler le graphe qui serait obtenir par la méthode combinatoire.

C'est d'après lui que l'on décidera :

- quels essais on fera en premier.
- quels résultats on va archiver.
- si on fera tel essai particulier.

Sa justification est purement heuristique : la définition que l'on a choisi sera considérée satisfaisante si elle conduit à un programme performant. Tout le programme repose sur cette notion d'intérêt. Comme il est libre de choisir ce qu'il va démontrer, il faut lui donner un but qui est d'obtenir des résultats ayant un intérêt aussi élevé que possible. Chez le mathématicien, il y a souvent une partie de l'intérêt qu'il affecte à un résultat qui est d'origine sémantique, liée à l'interprétation qu'il en donne. Ce n'est ici pas possible.

L'intérêt global va être la somme de trois intérêts que nous allons étudier successivement :

- intérêt syntaxique.
- intérêt généalogique.
- intérêt historique.

b - Intérêt syntaxique

Cet intérêt dépend uniquement de la structure de la suite de symboles qui constitue l'énoncé du résultat. Il est indépendant de l'axiomatique étudiée.

- Cas des théorèmes. Il est pour ceux-ci assez secondaire : son évaluation, très grossière, n'intervient qu'assez peu dans l'intérêt total du théorème. On ne tient compte que du nombre de connectives et l'on prend une formule : $I=I_o+Ab+Bu$ où u et b sont les nombres de connectives unaires et binaires figurant dans l'énoncé. A et B sont des coefficients négatifs. Si le théorème a peu de connectives on majore I d'une constante C pour favoriser davantage les théorèmes ayant très peu de connectives et qui sont en général importants.

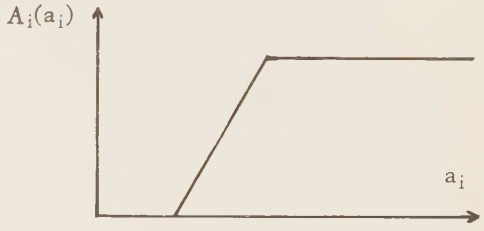
- Cas des productions à un antécédent. Pour celles-ci, il est capital. Il est calculé par une formule du type

$$I = \sum_i A_i(a_i)$$

Les A sont des fonctions données au programme. a_i est la valeur qu'a le paramètre i pour la production étudiée. Nous allons les énumérer plus loin.

Quand le paramètre prend pour valeurs des entiers, on a utilisé trois types de fonction A_i.

- linéaire (un ou deux segments de droite).
- parabolique pour le cas fréquent où il y a un optimum pour une valeur intermédiaire de a_i.
- linéaire avec écrêtage pour le cas où, à partir d'un certain niveau, une augmentation de a_i ne change plus l'intérêt de la production.



Le choix de la nature des paramètres est purement heuristique. Au départ on en a pris un certain nombre en s'aidant du bon sens. On a examiné les cas où l'intérêt calculé était en désaccord avec l'intuition. On a alors cherché quel nouveau paramètre pourrait remédier à cette difficulté. On est arrivé finalement à 21 paramètres qui peuvent se classer en 2 groupes : ceux ou chaque partie de la production (partie 1 : antécédent; partie 2 : résultat) est considérée séparément et ceux où l'on tient compte des deux parties.

- Paramètres dépendant d'une seule partie :

  - 1(2) : nombre de connectives binaires de la partie 1(2)

  - 3(4) : nombre de connectives unaires de la partie 1(2)

Exemple : $\supset\supset pqr \Rightarrow \supset qr$. $a_1=2$; $a_2=1$; $a_3=a_4=0$.

  - 5(6) : nombre de variables répétées dans 1(2). Une variable compte pour n-1 si elle figure n fois

  - 7(8) : variables différentes de 1(2). Une variable qui apparait plusieurs fois ne compte que pour 1.

Exemple : $\supset pq \Rightarrow \supset\supset rp \supset rq$. $a_5=0$; $a_6=1$; $a_7=2$; $a_8=3$.

$\supset\supset\supset\supset pqppr \Rightarrow r$. $a_5=2$; $a_6=0$; $a_7=3$; $a_8=1$.

Remarquons que l'on a : $a_1+1=a_5+a_7$ et $a_2+1=a_6+a_8$

  - 9(10) : connectives différentes de 1(2). C'est le nombre de connectives différentes (unaires et binaires) qui figurent dans une partie.

Exemple : $p\Rightarrow\supset qp$ $a_9=0$; $a_{10}=1$.

$\supset pq\Rightarrow\supset\sim q\sim p$. $a_9=1$; $a_{10}=2$.

$\supset pq\Rightarrow\supset\supset pq\supset rq$. $a_9=1$; $a_{10}=1$.

$\supset p\wedge qr\Rightarrow\supset pr$. $a_9=2$; $a_{10}=1$.

- Paramètres dépendant des deux parties :

- 11(12) : variables existant dans 1(2) et pas dans 2(1). Une variable apparaissant plusieurs fois dans 1 et pas dans 2 ne compte que pour 1.

Exemple : $\supset pq\Rightarrow\supset\supset rp\supset rq$. $a_{11}=0$; $a_{12}=1$.

$\supset\supset pqr\Rightarrow\supset qr$. $a_{11}=1$; $a_{12}=0$.

- 13 : nombre de variables apparaissant dans 1 et pas dans 2 plus nombre de variables apparaissant dans 2 et pas dans 1, chacune étant comptée autant de fois qu'elle occurre. On veut pouvoir pénaliser les productions qui modifient trop les théorèmes.

Exemple : $\supset pq\Rightarrow\supset\supset rp\supset rq$. $a_{13}=2$.

$p\Rightarrow\supset q\supset rq$. $a_{13}=4$.

- 14 : nombre de connectives binaires de 1 moins leur nombre dans 2. On se rapproche de la deuxième définition de la force. Il est préférable que ce nombre soit positif, mais il ne faut pas qu'il le soit trop pour que la production ne modifie pas trop les théorèmes auxquels on l'applique. L'expérience montre en effet que cela a en général un effet nocif. D'où une fonction A_{14} d'allure parabolique.

- 15 : analogue au précédent pour les connectives unaires.

Exemple : $\supset p\sim\sim q\Rightarrow\supset p\supset rq$. $a_{14}=-1$; $a_{15}=2$.

- 16 : $a_{16}=|a_{14}|+|a_{15}|$. Toujours pour éviter les grandes modifications. Une production qui enlève 2 connectives binaires et ajoute 2 unaires n'est pas très intéressante. a_{14} et a_{15} auraient 2 effets antagonistes et on ne pénaliserait pas la production.

- 17 : somme des valeurs absolues des différences des nombres d'occurrences dans 1 et dans 2 prises pour chaque connective. Dans a_{16} on les prenait par groupe unaire et binaire.

Exemple : $\wedge p\sim\sim q\Rightarrow q$. $a_{16}=a_{17}=3$.

$\supset pq\Rightarrow\supset\supset rp\supset rq$. $a_{16}=a_{17}=2$.

$\equiv pq\Rightarrow\supset\sim p\sim q$. $a_{16}=2$; $a_{17}=4$.

$\equiv pq\Rightarrow\supset pq$. $a_{16}=0$; $a_{17}=2$.

- 18(19) : connectives existant dans 1(2) et pas dans 2(1). C'est l'analogue de a_{11} et a_{12} pour les connectives.

Exemple : $p\Rightarrow\supset qp$. $a_{20}=0$; $a_{21}=1$.

$\supset pq\Rightarrow\supset\supset rp\supset rq$. $a_{20}=a_{21}=0$.

$\supset pq\Rightarrow\supset\sim q\sim p$. $a_{20}=0$; $a_{21}=1$.

$\equiv pq\Rightarrow\supset pq$. $a_{20}=a_{21}=1$.

- 20 : le type : c'est la force prise selon la première définition

fort : $\supset\supset pqr\Rightarrow\supset qr$

faible : $p \Rightarrow \supset qp$

égal : $\supset p \sim q \Rightarrow \supset q \sim p$

spécial : $\supset\supset pqr \Rightarrow \supset \sim pr$

Remarquons que les valeurs du paramètre a_{20} ne sont pas des entiers. Les productions les plus intéressantes sont les productions fortes.

- 21 : nombre de parties neutres. Nous allons préciser ce que l'on entend par partie neutre. On a voulu formaliser l'idée suivante : il est intéressant d'avoir une production où certaines parties de l'antécédent se retrouvent telles que dans le résultat. Cela permet d'agir localement sur les théorèmes et le résultat de la déduction sera très proche du point de départ.

Exemples : $\supset\supset pqr \Rightarrow \supset qr$ ne changera pas la partie Q de tout théorème de la forme $\supset PQ$ et agira sur la partie P comme le ferait la production $\supset pq \Rightarrow q$.

$\supset p \supset\supset qrs \Rightarrow \supset p \supset rs$ fait la même opération sur la partie Q d'un théorème du type $\supset P \supset QR$ etne modifie ni P ni R.

Tout ceci est lié à la règle de substitution. Dire que l'on agit sur une partie de théorème avec la production $\supset pq \Rightarrow q$ revient à substituer dans le théorème q à $\supset pq$.

Par définition nous dirons qu'une variable est une partie neutre d'une production si :

- elle apparait une fois et une seule dans chaque partie.
- elle y apparait en des emplacements correspondants.

Nous allons définir ce que l'on entend par emplacements correspondants de la production $A \Rightarrow B$. u et b désigneront une connective unaire et binaire, Q et R des termes quelconques, $\alpha, \beta, \gamma, \delta$ des suites de symboles quelconques.

1 - Si A est up, B doit être up
2 - Si A est bpQ, B doit être bpR
3 - Si A est bQp, B doit être bRp
4 - Si A est $\alpha up \beta$, B doit être $\gamma up \delta$, les suites $\alpha, \beta, \gamma, \delta$ devant être telles que p occupe des emplacements correspondants dans $\alpha p \beta$ et $\gamma p \delta$.
5 - Si A est $\alpha bpQ \beta$, B est $\gamma bpR \delta$
6 - Si A est $\alpha bQp \beta$, B est $\gamma bRp \delta$

$\alpha, \beta, \gamma, \delta$ étant toujours liés par la même relation.

Appliquons cette définition récursive à quelques exemples :

- $\supset\supset pqr \Rightarrow \supset qr$. r est une partie neutre : on applique 3 avec $Q = \supset pq$; $R = q$; $b = \supset$.

- $\supset pq \Rightarrow \supset p \supset rq$. p est une partie neutre : on applique 2 avec $Q = q$; $R = \supset rq$; $b = \supset$.

- $\supset p \supset qr \Rightarrow \supset q \supset pr$. r est une partie neutre. Si on applique 6 on a :

$\alpha = \supset p \quad \beta = \Lambda \quad Q = q \quad b = \supset$
$\gamma = \supset q \quad \delta = \Lambda \quad R = p$

et dans $\supset pr \Rightarrow \supset qr$, r est une partie neutre d'après 3.

- $\supset p \supset\supset qrs \Rightarrow \supset p \supset rs$. p et s sont des parties neutres.

pour p d'après 2
pour s on applique 6, puis 3.

- $\supset p \supset pq \Rightarrow \supset pq$. Ni p, ni q ne sont des parties neutres :

p apparait 2 fois dans l'antécédent

q n'apparait pas dans des emplacements correspondants.

- ≡pq⇒Cpq. ni p ni q ne sont des parties neutres. Les connectives diffèrent et on ne peut appliquer ni 2 ni 3.

- CCpqr⇒C~pr. r est une partie neutre d'après 3.

En résumé l'introduction des parties neutres a pour intérêt de compenser l'absence de possibilité de démontrer le théorème de substitution. L'avantage est que c'est très général et est utile même quand le théorème de substitution n'est que partiellement valable. Par contre il faut mémoriser toutes les productions correspondantes : si l'on veut pouvoir substituer P à Q et inversement, il faut :

P⇒Q	Q⇒P
CpP⇒CpQ	CpQ⇒CpP
CPp⇒CQp	CQp⇒CPp
CpCqP⇒CpCqQ	CpCqQ⇒CpCqP

A titre d'exemple donnons l'intérêt calculé par le programme pour diverses productions. L'intérêt n'est évidemment défini qu'à une constante additive près

CCpqr⇒Cqr	4800
CpCpq⇒Cpq	4550
CpCqr⇒CqCpr	4500
C~p~q⇒Cqp	4400
CpCCqrs⇒CpCrs	4350
CCpqp⇒p	4050
Cpq⇒C~q~p	3650
≡~~pq⇒Cpq	3500
~CpCqr⇒CCprs	3300
CpCqr⇒CpC≡sqr	3050
C~~≡pqp⇒C~~≡pqq	2800
CCCpqpCpq⇒CCCpqpq	2400
~p⇒CqCpr	2100
p⇒C~~~pq	1800
Cp~~q⇒CpCCqrr	1300
p⇒CCp~~qq	900
CCCCpqpCrps⇒Cts	-150
CpCCqCqrCqr⇒CsCCqCqrCqr	-500
CCCpCqqCrCCqqs⇒Cts	-1100
CCpCCCqrqCsqt⇒t	-1700
|||p|qrs|t|tt⇒||s|u|uu|v|wx	-2650

Assez paradoxalement, le programme a moins de finesse qu'un mathématicien pour classer les productions très intéressantes. Mais il arrive à départager de façon efficace des productions qui nous apparaissent également dénuées d'intérêt. Cette propriété est fondamentale dans les axiomatiques très difficiles où l'on doit au début démontrer des résultats assez peu intéressants pour pouvoir arriver ensuite à trouver des résultats vraiment intéressants.

A chaque production A⇒B on fait en réalité correspondre deux intérêts : l'un est évalué comme on vient de le dire, l'autre est celui de la production B⇒A. En effet dans certains cas une production opère comme si elle était renversée :

p⇒⊃qp+⊃p⊃qr,⊃pq⇒⊃pr⟷⊃qr,⊃pq⇒⊃pr

et tout se passe comme si c'était la production ⊃qp⇒p qui agissait sur le premier antécédent de la deuxième production. Il en est de même dans les démonstrations de conjectures. C'est pourquoi on classe les productions en deux groupes : les "fortes" et les "faibles", l'archivage dans le deuxième cas étant commandé par l'intérêt de la production inverse. Ces intérêts ne sont d'ailleurs différents que par suite d'une différence des valeurs des paramètres a_{14}, a_{15} et a_{20} qui correspondent justement aux deux premières définitions de la force que l'on a étudiées en 3B1.

Remarque 1 - Une production où l'antécédent est identique au résultat est considérée comme sans intérêt.

Exemple : ⊃⊃pqr⇒⊃⊃pqr

Remarque 2 - En aucun cas on n'a spécifié la nature des connectives sur lesquelles on opérait. On a seulement supposé que l'on pouvait tester leur nature. Ce fait rend le programme très général et lui permet de traiter aussi bien une axiomatique où figure le Sheffer's stroke qu'une axiomatique qui possède l'implication et la négation.

- Cas des productions à deux antécédents

La méthode de calcul de leur intérêt est certainement le point faible du programme : d'une part elle manque de généralité et est trop axée vers les axiomatiques de la logique des propositions ; d'autre part elle manque d'efficacité en ne voyant pas l'intérêt de certains résultats pourtant utiles.

On considère arbitrairement trois types intéressants, toute production n'appartenant pas à un de ces types et jugée sans intérêt.

.1 - type déductif. L'un des antécédents se réduit à une variable, soit p ; le résultat se réduit aussi à une variable, soit q. p et q figurent dans l'autre antécédent. On peut l'écrire : p,f(p,q)⇒q.

Exemple : p,⊃pq⇒q
p,|p|qr⇒r
p,⊃~q~p⇒q

mais les productions suivantes ne sont pas de ce type :

~p,⊃~qp⇒q : aucun antécédent ne se réduit à une variable
p,⊃pq⇒⊃rq : le résultat n'est pas une variable
p,~p⇒q : q ne figure pas dans un antécédent.

On applique automatiquement à toute production de ce type les deux MMMthéorèmes :

a,b⇒c⇝b⟶a⇒c
a,b⇒c⇝p⇒b⟶p,a⇒c

ce qui crée les nouveaux MMthéorèmes :

f(p,q)⟶p⇒q
p⇒f(q,r)⟶p,q⇒r

Exemple : si f(p,q) est |p|qr, on a :
|p|qr⟶p⇒r
p⇒|q|rs⟶p,q⇒s

2 - type substitutif. La production A,B⇒C est de ce type si l'une des productions au moins : A⇒C ou B⇒C comporte une partie neutre, l'antécédent et le résultat ayant alors une seule connective

Exemple : ⊃p⊃qr,⊃pq⇒⊃pr car ⊃pq⇒⊃pr à la partie neutre p.

⊃pq,⊃qr⇒⊃pr l'est doublement

⊃pq⇒⊃pr à la partie neutre p

⊃qr⇒⊃pr à la partie neutre r

Mais ⊃pq,⊃qp⇒≡pq n'est pas de ce type, car ⊃pq⇒≡pq n'a pas de partie neutre.

On applique à ces productions le MMMthéorème : a,b⇒c⇝a⟶b⇒c si b⇒c a une partie neutre.

Exemple : - ⊃p⊃qr,⊃pq⇒⊃pr donnera naissance au MMthéorème : ⊃p⊃qr⟶⊃pq⇒⊃pr.

- ⊃pq,⊃qr⇒⊃pr donnera naissance à 2 MMthéorèmes : ⊃pq⟶⊃qr⇒⊃pr et ⊃pq⟶⊃rp⇒⊃rq.

Appliqué à ⊃p⊃qp, le premier donne ⊃⊃pqr⇒⊃qr et le second ⊃pq⇒⊃p⊃rq.

3 - type utilisé pour les démonstrations de conjectures. Soit la production a,b⇒c et B_a et U_a les nombres de connectives binaires et unaires de la partie a.

On pose les conditions :

$B_a \leqslant B_c ; B_b \leqslant B_c$

$B_a + U_a \leqslant B_c + U_c ; B_b + U_b \leqslant B_c + U_c$

En effet, on veut avoir des résultats plus faibles quand on l'applique pour démontrer des conjectures. (ceci dans un sens proche de la deuxième définition de la force).

A chaque conjecture, on associera par cette production deux nouvelles conjectures qui seront plus simples, donc plus faciles à démontrer que la conjecture initiale. Cette dernière sera démontrée quand on aura démontré les deux nouvelles conjectures.

Exemples : -⊃pq,⊃qp⇒≡pq est de ce type. A la conjecture ≡∧∧pqr∧p∧qr, il va substituer les deux conjectures : ⊃∧∧pqr∧p∧qr et ⊃∧p∧qr∧∧pqr.

- ⊃pq,⊃pr⇒⊃p∧qr est également de ce type. A la conjecture ⊃∧∧pqr∧p∧qr il va substituer : ⊃∧∧pqrp et ⊃∧∧pqr∧qr.

A toute production de ce type, on applique le MMMthéorème : a,b⇒c⇝c⇒p⟶a,b⇒p.

Avec p,q⇒∧pq, on obtiendra ∧pq⇒r⟶p,q⇒r qui établit un lien entre le "et" de la langue et celui de la Mlangue.

Remarque - On élimine naturellement toute production où un des antécédents est identique au résultat: ⊃pq,⊃qr⇒⊃pq. Dans certains cas on se ramène à une production à un antécédent en utilisant les MMthéorèmes 2 et 4 exposés en 2D2a. On regarde systématiquement si on n'est pas dans un de ces cas.

On voit bien que l'on a apporté de sérieuses restrictions au programme. Ces trois types sont très spécialisés vers la logique des propositions, le premier étant inspiré de l'utilisation du modus ponens, le deuxième visant à suppléer le fait que l'on n'utilise pas le théorème de substitution. La définition de ces types vise à avoir un MMthéorème du type a⟶b⇒c tel que b⇒c ait des chances d'être intéressant et qu'il y ait une certaine parenté entre a et b⇒c.

Mais à ce manque de généralité se superpose le reproche du manque d'efficacité. Le programme

considère sans intérêt des productions qui sont importantes, quand on a rien de mieux :

$\supset pq, \sim \wedge qr \Rightarrow \sim \wedge rp$

$\supset pq, \supset p \sim q \Rightarrow \sim p$

et ceci est la principale cause de la faiblesse du programme quand il traite une axiomatique où de telles productions jouent un rôle important. Malheureusement, on ne voit pas comment on pourrait découvrir l'intérêt de ces productions sans admettre en même temps un grand nombre d'autres productions inutiles.

c - *Intérêt généalogique*

C'est une fonction de l'intérêt des Mthéorèmes auxquels le résultat donne naissance quand on lui applique les MMthéorèmes. Il est fondamental pour déterminer l'intérêt des théorèmes, auxquels on applique les MMthéorèmes du type $a \longrightarrow b \Rightarrow c$. Il est beaucoup moins important pour les productions à un antécédent à qui on applique les MMthéorèmes du type $a \Rightarrow b \longrightarrow c, d \Rightarrow e$, car peu de productions à 2 antécédentes sont considérées comme intéressantes. On n'évalue pas d'intérêt généalogique pour les productions à deux antécédents (le programme ne démontre pas de MMthéorème que l'on puisse leur appliquer).

Cet intérêt est vraiment proche de notre intuition : si on étudie $\supset\supset p \supset qr \supset q \supset pr$, on le considère comme très intéressant, mais ce n'est pas pour sa forme. C'est parce qu'on peut lui appliquer le MMthéorème $\supset pq \longrightarrow p \Rightarrow q$. On a alors la production $\supset p \supset qr \Rightarrow \supset q \supset pr$ qui, elle, est intéressante par la façon dont elle transforme les suites de symboles. Si $\supset pq \longrightarrow p \Rightarrow q$ n'existait pas, $\supset\supset p \supset qr \supset q \supset pr$ aurait beaucoup moins d'intérêt. De même prenons $\supset p \supset\supset pqq$ qui donne $p \Rightarrow \supset\supset pqq$ qui donne à son tour $p, \supset pq \Rightarrow q$ qui est le modus ponens. Son intérêt va remonter à $p \Rightarrow \supset\supset pqq$ et de là au théorème. Mais aucun de ces derniers n'avait une forme particulièrement intéressante.

Voyons un peu la façon dont le programme opère. Quand il a un théorème nouveau il lui applique les MMthéorèmes qu'il connait et va en déduire un certain nombre de productions à un antécédent. Il va à son tour leur appliquer des MMthéorèmes et obtiendra des productions à deux antécédents. Ayant évalué leur intérêt syntaxique, il en déduira l'intérêt généalogique des productions à un antécédent "mères". Il va l'additionner à leur intérêt syntaxique ce qui permettra de calculer l'intérêt généalogique du théorème de départ.

L'intérêt généalogique est une fonction linéaire de l'intérêt des descendants les plus intéressants.

On ne peut pas justifier cette méthode de façon rigoureuse, car elle est de nature heuristique. La seule justification possible est de montrer qu'elle donne un classement en accord avec ce qu'attend notre intuition. Notons que l'intérêt d'un énoncé peut varier d'une axiomatique à l'autre, car l'intérêt généalogique dépend des descendants obtenus par action des MMthéorèmes. Et les MMthéomes, qui sont démontrés par le programme, varient d'une axiomatique à l'autre.

Donnons à titre d'exemple les valeurs de l'intérêt syntaxique plus généalogique données par le programme à un certain nombre de théorèmes de l'axiomatique 1.

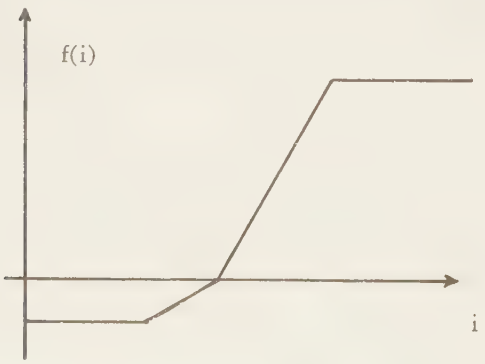
$\supset\supset pq \supset\supset qr \supset pr$	6690
$\supset\supset p \supset qr \supset q \supset pr$	6134
$\supset p \supset\supset pqq$	5891
$\supset\supset p \supset qr \supset\supset pq \supset pr$	5786
$\supset \wedge pqp$	5421

Cc~~pp	5360
Cc~CpqCpq	5197
CCpc~qCqc~p	4895
Ce=pqCpq	4610
CCpCpqCpq	4595
CCpqCc~qc~p	4301
CCc~pCc~qpCc~pq	3980
CpCqq	3938
Ce=~pqCpc~q	3739
CpCCqCprCqr	3542
Cc~CppCpq	3545
CCCpqpCCpqq	3094
Cc~c~pCqp	2915
CpCqCrr	2749
CpCCpCpqq	2500
Cc~~~~pp	2115
CpCc~~~~qq	1869
Cc~Ac~pc~pp	1345
Cc~CApqpr	778
CpCCpCc~qrCc~rq	140

Remarque - On voit ici une des raisons pour lesquelles on a introduit les MMthéorèmes. Cette raison est heuristique : cela permet d'évaluer avec plus de précision l'intérêt des théorèmes. C'est absolument indispensable tant que l'on se refuse à faire de la sémantique : si l'on ne donne pas au théorème un lien avec le monde extérieur, on ne voit pas comment on pourrait définir son inétrêt d'une autre façon que celle que l'on a choisie. Le comptage des différentes sortes de connectives et de variables apparaissant dans le théorème ne permet pas d'avoir une idée de l'intérêt.

d - Intérêt historique

Il s'agit de modifier, en tenant compte de l'expérience acquise, l'intérêt calculé à l'aide des deux méthodes précédentes. Cela va servir à corriger les erreurs d'évaluation que l'on aurait pu faire. On pourra éliminer des résultats surévalués et revaloriser d'autres résultats mal jugés initialement. Pour cela on va tenir compte de l'intérêt des résultats qu'il contribue à créer : pour un théorème par action des productions, pour une production par action sur les théorèmes.



Pour chaque résultat nouveau, on ajoute à l'intérêt des deux résultats dont il est issu une fonction de son intérêt. Si i est l'intérêt du résultat trouvé, on ajoutera f(i) qui a l'allure ci-contre aux intérêts des résultats "pères". Il y a écrêtage aux deux extrémités pour éviter d'avoir une réaction trop forte causée par un seul résultat. Les pentes différentes viennent de ce qu'il est normal d'avoir plus de résultats inintéressants que de résultats intéressants.

La correction qu'apporte l'intérêt historique est en général assez faible. Son rôle est très secondaire par rapport aux deux précédents. A titre d'exemple nous allons donner l'évolution de l'intérêt total au cours d'une exploitation de l'axiomatique 1 en choisissant les résultats qui ont le plus varié. L'évolution peut être très différente d'une axiomatique à l'autre, car elle dépend des autres résultats que l'on a trouvé.

énoncé	valeur initiale	valeur finale
⊃⊃p⊃qr⊃q⊃pr	6134	6584
⊃⊃p⊃qr⊃⊃pq⊃pr	5786	6406
⊃~~pp	5360	9730
⊃⊃pqr⇒⊃qr	4800	9100
⊃⊃p⊃pq⊃pq	4595	5705
⊃p⊃q~q⇒⊃p~q	4250	3370
⊃⊃p~⊃qr⊃pq	4150	3920
⊃⊃~⊃~pqq⊃~qp	4110	3760

Remarque - Très souvent un théorème se redonne lui-même ou donne le même résultat quand on lui applique diverses productions

Exemples 1 - ⊃pp+⊃⊃pq⊃rp⇒⊃rp⊣↦⊃pp
⊃pp+⊃⊃pq⊃qr⇒⊃qr⊣↦⊃pp

2 - ⊃⊃⊃ppqq+⊃⊃⊃pq⊃rq⊃qs⇒⊃qs⊣↦⊃pp
⊃⊃⊃ppqq+⊃⊃⊃pqrq⇒⊃pq⊣↦⊃pp
⊃⊃⊃ppqq+⊃⊃ppq⇒q⊣↦⊃pp
⊃⊃⊃ppqq+⊃⊃p⊃qqr⇒r⊣↦⊃pp
⊃⊃⊃ppqq+⊃⊃p⊃qr⊃sq⇒⊃sq⊣↦⊃pp
⊃⊃⊃ppqq+⊃⊃pqp⇒p⊣↦⊃pp

Il serait anormal d'augmenter l'intérêt des résultats initiaux à chaque fois. Le résultat trouvé n'est pas nouveau et ces dérivations ne font que traduire que ce que l'on a dérivé est en quelque sorte "compris" dans un des résultats de départ. Aussi on ne modifie pas l'intérêt si l'on se trouve dans ce cas.

e - Conclusion

L'intérêt d'un résultat est la somme de trois intérêts :

- l'intérêt syntaxique qui dépend de la forme du résultat.
- l'intérêt généalogique qui est une fonction de l'intérêt des Mthéorèmes que l'on obtient par application des MMthéorèmes.
- l'intérêt historique qui est une fonction de l'intérêt des résultats obtenus au cours des essais où il participe.

Pour un même énoncé, le premier ne variera pas d'une axiomatique à l'autre. Les deux derniers dépendent des autres résultats démontrés et n'ont pas forcément la même valeur dans deux axiomatiques différentes. Les deux premiers sont calculés une fois pour toute au moment où l'on a démontré le résultat. Le troisième est une correction qui tient compte de l'expérience acquise dans les essais faits ultérieurement.

La distinction entre intérêt généalogique et historique est assez subtile. Quand on a un résultat nouveau, on ne peut pas l'essayer immédiatement avec tout ce que l'on connait, car il se peut qu'il soit sans intérêt. On l'essaye donc avec les MMthéorèmes qui ne sont pas très nombreux et qui font une analyse beaucoup plus profonde de son intérêt. Il s'agit quand même d'un artifice pour suppléer à la véritable mesure de l'intérêt que l'on peut faire dans une interprétation. Comme les axiomatiques étudiées sont logiques, les théorèmes sont l'expression de lois logiques et l'application des MMthéorèmes permet d'établir un parallèle entre la langue et la Mlangue qui est, elle aussi, logique. La mesure que l'on a donné pour l'intérêt est peut être liée trop étroitement aux systèmes formels ayant une interprétation logique. Elle semble pourtant correspondre à notre propre mécanisme d'évaluation pour autant que l'on puisse en juger.

Insistons enfin sur l'importance de cette mesure de l'intérêt : toutes les heuristiques vont reposer dessus. S'il est mal calculé, le programme démontrera des théorèmes dénués d'intérêt.

3B3 - Elimination de résultats par les résultats déjà démontrés

a - Introduction

A côté de l'élimination des résultats qui ont intérêt trop faible, il y a une autre sorte d'élimination pour les résultats qui ne sont pas utiles, car on les a déjà sous une forme plus forte ou équivalente.

Si l'on a démontré que Cpp est un théorème, on ne va évidemment pas garder les théorèmes :

$CCpqCpq$; $C{\sim}p{\sim}p$; $C{\sim}Cpq{\sim}Cpq$;....

qui s'en déduisent par substitution, même si le programme leur assigne un intérêt élevé. L'élimination que l'on va étudier dans cette partie est indépendante de la structure du résultat. Nous distinguerons le cas où l'on peut donner une justification rigoureuse de cette élimination (par exemple celle de $C{\sim}p{\sim}p$ quand on a Cpp) du cas où elle repose sur des considérations heuristiques.

En reprenant la notation introduire en 3B1, nous écrirons a<b, a et b étant des suites de symboles, si l'on peut passer de b à a par une substitution qui n'est pas une permutation des noms de variables. Si c'est seulement une permutation des noms de variables nous écrirons a=b. $a \leqslant b$ est une abréviation de : a<b ou a=b.

Exemples : $C{\sim}p{\sim}p \subset Cpp$

$p \Rightarrow Cqp > p \Rightarrow CCqpp$

$p \Rightarrow Cqr \longrightarrow p, q \Rightarrow r < {\sim}p \Rightarrow Cqr \longrightarrow {\sim}p, q \Rightarrow r$

$CpCqr \Rightarrow CqCpr = CqCrp \Rightarrow CrCqp$

Remarque - On considère comme égales deux productions à deux antécédents qui deviennent égales en changeant l'ordre de leurs antécédents : $Cpq, Cqr \Rightarrow Cpr = Cpq, Crp \Rightarrow Crq$. Par contre, on considère comme différentes : $a \longrightarrow b \Rightarrow c$ et $b \longrightarrow a \Rightarrow c$.

b - Eliminations que l'on peut justifier rigoureusement

1 - Pour tous les résultats. Quand on a un nouveau résultat A, on regarde :

- s'il n'existe pas déjà sous une forme égale B, auquel cas on élimine A. Cela se produit quand il y a ambiguité.

- s'il n'existe pas un résultat C tel que C>A. On élimine alors A.

- s'il n'existe pas un résultat D tel que D<A. On élimine alors D et on poursuit l'étude de A.

Exemple : Si l'on vient de démontrer ⊃pp et que l'on avait déjà archivé ⊃~p~p, on efface ce dernier.

Tout ceci est lié à la substitution : on ne garde que le résultat qui a l'énoncé le plus fort.

2 - Pour les théorèmes. Soit un nouveau théorème A.

- s'il existe B, antécédent d'une production, tel que A⩾B, on remplace une production à un antécédent par un théorème : son résultat et une production à deux antécédents par une production à un antécédent : si l'autre antécédent, alors le résultat.

Ceci est évident d'après la définition des productions, car si A⩾B, cela signifie que l'antécédent B est un théorème.

Exemples : Si l'on a la production : ⊃~p~p⇒⊃p⊃qp et que l'on démontre ⊃pp, on en déduit que ⊃p⊃qp est un théorème.

- Si l'on a la production : p,⊃pp⇒⊃qp, on en déduit la production p⇒⊃qp si l'on a démontré ⊃pp.

- S'il existe B, résultat d'une production, tel que A>B, la production est sans intérêt puisqu'elle ne peut donner que A ou un résultat se déduisant de A par substitution.

Exemple : ⊃⊃⊃pqr⊃sq⇒⊃q⊃pq est sans intérêt à partir du moment où l'on a démontré ⊃p⊃qp. De tels cas sont assez fréquents.

De même p,q⇒⊃~p~p est sans intérêt si l'on a ⊃pp.

- S'il existe le MMthéorème B⟶C⇒D et que A>B, le MMthéorème est sans intérêt, car C⇒D est une production et l'application du MMthéorème ne peut donner naissance qu'à des productions plus faibles.

3 - Pour les productions à un antécédent quand on a une nouvelle production A⇒B, on regarde :

- Si A⩽C, C étant un théorème, B est alors un théorème et la production est sans intérêt.

- Si B⩽C, C étant un théorème, la production est sans intérêt.

- Si A⇒B⩾C⇒D et s'il existe une production à deux antécédents : C,E⇒D, cette production à 2 antécédents est sans intérêt : On peut avoir le même résultat en faisant moins d'hypothèses.

Exemple : p,q⇒⊃qp est sans intérêt si l'on a p⇒⊃qp.
p,⊃qr⇒~qp est également sans intérêt si l'on a p⇒⊃qp.

- Si A⇒B⩾C⇒D et s'il existe un MMthéorème : C⟶E⇒D ou E⟶C⇒D, le MMthéorème est sans intérêt.

Exemple : ∧pq⟶q⇒p est sans intérêt si l'on a ∧pq⇒p.

4 - Pour les productions à deux antécédents A,B⇒C, on regarde :

- Si A⩽D ou B⩽D, D étant un théorème, on se ramène à une production à un antécédent : B⇒C ou A⇒C.

- Si C⩽D, D étant un théorème, la production est sans intérêt.

- Si $A \Rightarrow C \leqslant D \Rightarrow E$ ou $B \Rightarrow C \leqslant D \Rightarrow E$, $D \Rightarrow E$ étant une production à un antécédent, la production à deux antécédents est sans intérêt.

5 - Pour les MMthéorèmes du type $A \longrightarrow B \Rightarrow C$, on regarde :

- Si $A \leqslant D$, D étant un théorème, $B \Rightarrow C$ est une production.

- Si $A \Rightarrow C \leqslant D \Rightarrow E$ ou $B \Rightarrow C \leqslant D \Rightarrow E$, $D \Rightarrow E$ étant une production, le MMthéorème est sans intérêt.

Remarque 1 - On voit que le même phénomène peut conduire à deux recherches différentes selon le résultat qui apparaitra le premier. Prenons par exemple le cas où l'on a le théorème A et la production $B \Rightarrow C$ avec $A \geqslant B$. Si l'on a A en premier, on s'en apercevra au moment où l'on étudiera la production. Si l'on a d'abord $B \Rightarrow C$ on s'en apercevra au moment où l'on fait la recherche pour A. Dans les deux cas, on éliminera évidemment la production. Mais dans le premier on se contentera de ne pas l'archiver alors que dans le second il faudra l'enlever des endroits où elle est archivée.

Remarque 2 - On aurait pu rajouter deux autres types de recherches appartenant à la même famille. On ne l'a pas fait, car on les a jugées très peu probables.

- Si $A \Rightarrow B \geqslant C \Rightarrow D$ que l'on ait la production $A \Rightarrow B$ et le MMthéorème $C \Rightarrow D \longrightarrow E, F \Rightarrow G$. Le MMthéorème est alors sans intérêt et $E, F \Rightarrow G$ est une production.

Exemple : $p \Rightarrow \supset \sim qp \longrightarrow p, q \Rightarrow \wedge pq$. Si l'on démontre $p \Rightarrow \supset qp$, on en déduit la production $p, q \Rightarrow \wedge pq$.

- Si $A \Rightarrow B \geqslant C \Rightarrow D$ et si l'on a la production $C, D \Rightarrow E$, la production à deux antécédents peut être remplacée par la production $C \Rightarrow E$.

Il est en effet évident que si l'on a C on peut en déduire D, donc il suffit de faire seulement l'hypothèse que l'on a C pour pouvoir en déduire E. Ce n'est au fond que l'application des deux MMthéorèmes :

$a \Rightarrow b$; $c, b \Rightarrow d \longrightarrow a, c \Rightarrow d$ où l'on fait $a = c$

et $a, a \Rightarrow b \longrightarrow a \Rightarrow b$

qui conduisent à $a \Rightarrow b; a, b \Rightarrow c \longrightarrow a \Rightarrow c$

Exemple : On a : $p, \supset \sim \sim qp \Rightarrow \supset \sim pq$ et $p \Rightarrow \supset qp$. On en déduit $p \Rightarrow \supset \sim pq$ car si l'on a p on a $\supset qp$ donc $\supset \sim \sim qp$ (par substitution), donc $\supset \sim pq$ (grâce à la première production).

c - *Eliminations reposant sur des critères heuristiques.*

Les éliminations que l'on vient de décrire paraissent totalement justifiées, et l'on ne voit pas de cas où elles pourraient avoir un effet nuisible. Nous allons maintenant éliminer des résultats sur des critères beaucoup plus discutables, basés uniquement sur l'expérience. Ces éliminations porteront uniquement sur des Mthéorèmes et des MMthéorèmes. Elles n'auront en général pas d'effet nocif, car les résultats qu'elles auraient permis d'obtenir pourront être dérivés par d'autres moyens. Cela n'empêche pas que dans certains cas, cela aura un résultat malheureux. C'est une conséquence de l'utilisation de méthodes qui n'ont qu'une valeur heuristique. Dans la grande majorité des cas, cette élimination a eu un effet positif en évitant d'encombrer la mémoire et de faire des essais inutiles.

On n'élimine pas directement des théorèmes. Mais on le fait indirectement, car si on élimine les Mthéorèmes auxquels ils donnent naissance, leur intérêt généalogique va en être affecté.

1 - On a les productions A⇒B; C⇒D; E⇒F. A⇒B est une production forte au sens de la 1ère définition (A<B) et C⇒D une production faible (C>D).

S'il existe des substitutions telles que :

$A \Rightarrow B \geqslant a \Rightarrow b$

$C \Rightarrow D \geqslant b \Rightarrow d$

$E \Rightarrow F \leqslant a \Rightarrow d$

alors la production E⇒F sera considérée comme sans intérêt.

Ceci permet d'éliminer un grand nombre de productions que notre intuition juge, à juste titre, sans intérêt. Supposons que l'on ait les productions fortes :

$\wedge pq \Rightarrow p;\ \wedge pq \Rightarrow q;\ \sim\sim p \Rightarrow p$

et les productions faibles :

$p \Rightarrow \supset qp;\ p \Rightarrow \sim\sim p;\ p \Rightarrow \vee pq;\ p \Rightarrow \vee qp$

On voit que l'on a aussi les productions :

$\wedge pq \Rightarrow \supset rp;\ \wedge pq \Rightarrow \sim\sim p;\ \wedge pq \Rightarrow \vee pr;\ \wedge pq \Rightarrow \vee rp$

$\wedge pq \Rightarrow \supset rq;\ \wedge pq \Rightarrow \sim\sim q;\ \wedge pq \Rightarrow \vee qr;\ \wedge pq \Rightarrow \vee rq$

$\sim\sim p \Rightarrow \supset qp;\ \sim\sim p \Rightarrow \sim\sim p;\ \sim\sim p \Rightarrow \vee pq;\ \sim\sim p \Rightarrow \vee qp$

qui sont toutes sans intérêt.

Si vraiment on voulait en appliquer une, on a intérêt à appliquer d'abord la production forte qui donnera un théorème plus simple que celui de départ et à ce moment là voir s'il y a lieu d'appliquer la production faible. Souvent ces productions ont un intérêt syntaxique élevé et elles sont très nombreuses, ce qui rend leur élimination encore plus nécessaire.

2 - On a les productions A⇒B; C⇒D; E⇒F qui sont toutes les trois du type fort : A<B; C<D; E<F, et il existe des substitutions telles que :

$A \Rightarrow B \geqslant a \Rightarrow b$

$C \Rightarrow D \geqslant b \Rightarrow d$

$E \Rightarrow F \leqslant a \Rightarrow d$

La production E⇒F est alors considérée sans intérêt.

Là encore on juge inutile d'introduire un double moyen pour obtenir un résultat. Si le résultat de départ est assez intéressant pour que l'on juge utile de lui appliquer la production A⇒B, il est préférable (mais non certain) que le résultat obtenu avec cette production forte sera encore plus intéressant et qu'on lui appliquera C⇒D. Il est donc inutile de prévoir aussi le raccourci E⇒F.

Exemple 1 - Si on a les productions :

$\supset p \sim\sim q \Rightarrow \supset pq;\ \supset \sim\sim pq \Rightarrow \supset pq;\ \supset \sim\sim p \sim\sim q \Rightarrow \supset pq$

on éliminera la dernière.

Exemple 2 - Si l'on a :

$\supset \sim p \sim q \Rightarrow \supset qp;\ \supset\supset pqr \Rightarrow \supset qr;\ \supset \sim p \sim \supset qr \Rightarrow \supset rp$

on éliminera la dernière. En effet a⇒b est $\supset \sim p \sim \supset qr \Rightarrow \supset\supset qrp$ et b⇒d : $\supset\supset qrp \Rightarrow \supset rp$.

3 - On a les productions $A \Rightarrow B$; $C \Rightarrow D$; $E \Rightarrow F$ qui sont toutes les trois du type faible $A > B$; $C > D$; $E > F$ et il existe des substitutions telles que :

$A \Rightarrow B \geqslant a \Rightarrow b$

$C \Rightarrow D \geqslant b \Rightarrow d$

$E \Rightarrow F \leqslant a \Rightarrow d$

La production $E \Rightarrow F$ sera considérée comme sans intérêt.

C'est l'analogue du précédent pour les productions faibles. On peut faire comprendre l'utilité de ceci par un raisonnement parallèle appliqué aux démonstrations de conjectures au lieu des démonstrations directes : en effet les productions faibles sont surtout importantes pour les conjectures.

Exemple 1 - On a les productions :

$p \Rightarrow Cqp$; $Cpq \Rightarrow Cp{\sim}{\sim}q$; $p \Rightarrow Cq{\sim}{\sim}p$

on éliminera la dernière.

Exemple 2 - On a les productions :

$p \Rightarrow Cqp$; $CCpqr \Rightarrow CCpCpqr$; $p \Rightarrow CCqCqrp$

on éliminera encore la dernière.

4 - On a la production $A \Rightarrow B$ et les MMthéorèmes $C \longrightarrow D \Rightarrow E$ et $F \longrightarrow G \Rightarrow H$. $A \Rightarrow B$ est une production forte ($A < B$)

S'il existe des substitutions telles que :

$A \Rightarrow B \geqslant a \Rightarrow b$

$C \longrightarrow D \Rightarrow E \geqslant b \longrightarrow d \Rightarrow e$

$F \longrightarrow G \Rightarrow H \leqslant a \longrightarrow d \Rightarrow e$

on éliminera le MMthéorème $F \longrightarrow G \Rightarrow H$.

On veut éviter de créer trop de MMthéorèmes ce qui compliquerait le calcul de l'intérêt généalogique. Si on a le théorème T, on lui appliquera tout naturellement la production $A \Rightarrow B$ s'il est intéressant et on appliquera au résultat le MMthéorème $C \longrightarrow D \Rightarrow E$. Il est donc, en général, inutile de prévoir la nécessité d'appliquer $F \longrightarrow G \Rightarrow H$ qui aurait au plus le même effet.

Exemple 1 - On a la production : $C{\sim}p{\sim}q \Rightarrow Cqp$ et les MMthéorèmes : $Cpq \longrightarrow p \Rightarrow q$ et $C{\sim}p{\sim}q \longrightarrow q \Rightarrow p$

on va éliminer le deuxième MMthéorème.

Exemple 2 - On a la production : $Cp{\sim}{\sim}q \Rightarrow Cpq$ et les MMthéorèmes : $Cpq \longrightarrow p \Rightarrow q$ et $C{\sim}p{\sim}{\sim}Cqr \longrightarrow {\sim}p \Rightarrow Cqr$

On voit que :

$a \Rightarrow b$ est : $C{\sim}p{\sim}{\sim}Cqr \Rightarrow C{\sim}pCqr$

$b \longrightarrow d \Rightarrow e$ est : $C{\sim}pCqr \longrightarrow {\sim}p \Rightarrow Cqr$

$a \longrightarrow d \Rightarrow e$ est : $C{\sim}p{\sim}{\sim}Cqr \longrightarrow {\sim}p \Rightarrow Cqr$

et l'on élimine le deuxième MMthéorème.

Ces 4 cas sont très analogues et sont basés sur le principe suivant : si, quand on associe deux résultats tels que ou sous une forme plus faible (d'où les deux premiers $\geqslant$), on a un nouveau résultat égal ou plus fort (d'où le $\leqslant$ de la troisième ligne) qu'un troisième résultat, ce dernier est sans intérêt. Ces 4 cas conduisent à un grand nombre de recherches, car le résultat nouveau peut correspondre à l'un quelconque des trois résultats que l'on fait intervenir.

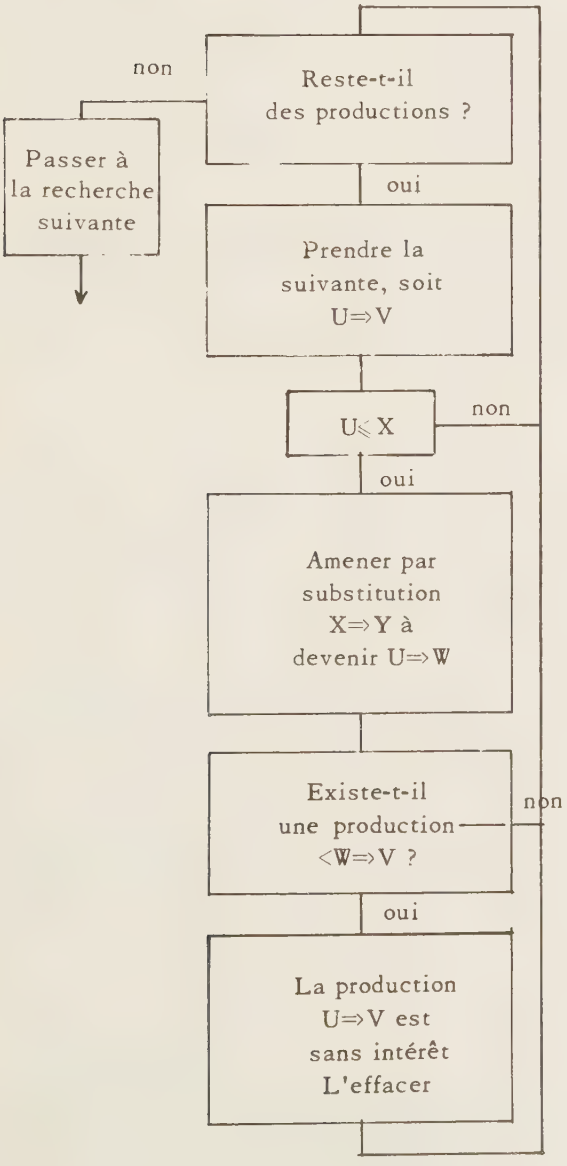
Supposons que l'on ait une nouvelle production X⇒Y qui est forte. Il faudra, pour traiter le 2è cas :

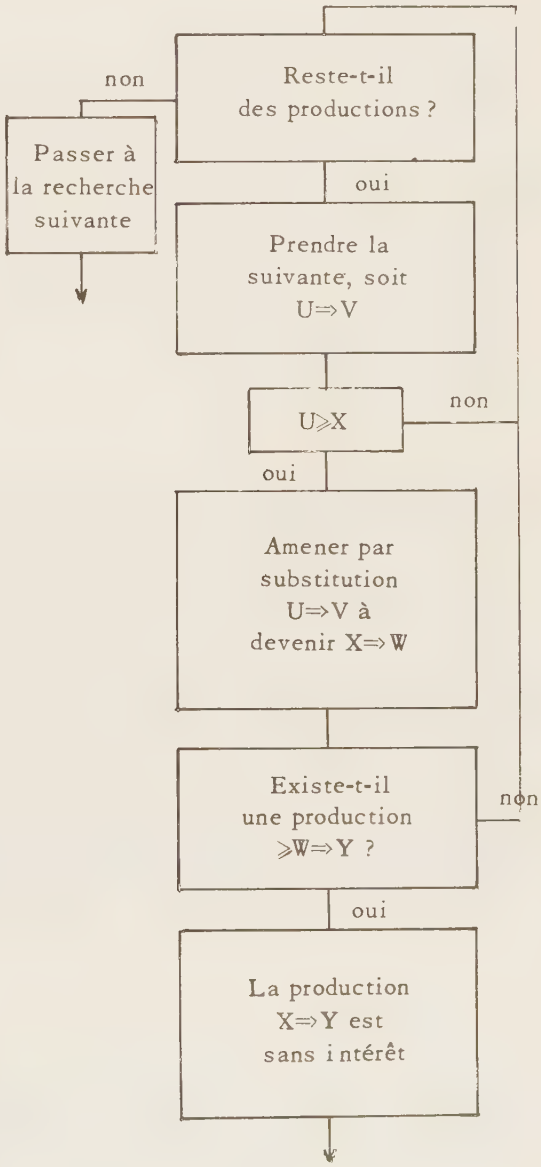
- chercher s'il existe une production U⇒V avec U⩽X. Si oui X⇒Y devient par substitution U⇒W. On cherche alors s'il existe une production ⩾ à W⇒V. Si oui, la production U⇒V est sans intérêt.

- chercher s'il existe une production U⇒V avec V⩽Y. Si oui X⇒Y devient par substitution W⇒V. On cherche alors s'il existe une production ⩾à U⇒W. Si oui, la production U⇒V est sans intérêt.

- chercher s'il existe une production U⇒V avec U⩾X. Si oui, U⇒V devient X⇒W par substitution. On cherche alors s'il existe une production ⩾ à W⇒Y. Si oui la production X⇒Y est sans intérêt. Bien entendu, comme on a supposé que l'on traitait le deuxième cas on ne fait les recherches que parmi les productions fortes. Pour cette production forte, il faudra faire également 2 recherches pour le 1er cas et une pour le 4è.

Donnons 2 organigrammes possible pour ces recherches (pour les 1er et 3è variétés du 2è cas).





Remarque 1 - Dans l'organigramme de droite, on arrête les recherches dès que l'on a vu que la production de départ est sans intérêt. Dans celui de gauche, la production sans intérêt était déjà archivée et il faut poursuivre les recherches, car cela amènera peut être à en éliminer d'autres. Par exemple dans l'axiomatique 3, on a les productions

$CCpqCqr \Rightarrow CsCqr$	$CCpqCqr \Rightarrow Cqr$
$CCpqp \Rightarrow Crp$	$CCpqp \Rightarrow p$
$CCCpqCrqCqs \Rightarrow CtCqs$	$CCCpqCrqCqs \Rightarrow Cqs$

La découverte de $p \Rightarrow Cqp$ va éliminer les trois productions de gauche.

Remarque 2 - On n'a pas précisé le contenu de la case : existe-t-il une production $\geqslant W \Rightarrow Y$? Il s'agit d'un sous programme très simple qui est également utilisé pour les techerches vues en b. On prend les productions connues et on regarde si par substitution on peut amener à la faire coïncideravec $W \Rightarrow Y$. La simplicité doit beaucoup à l'utilisation de la notation polonaise.

Le programme ainsi décrit prendrait beaucoup de temps, car si l'on a N productions, on a N^2 recherches ; Il y a en effet deux boucles : l'une pour chercher la production suivante et l'autre pour chercher s'il existe une production $\geqslant W \Rightarrow Y$. Pour gagner du temps, le programme a été très modifié de façon à faire des recherches simultanément pour plusieurs résultats et cela a été unedes raisons qui ont conduit à l'archivage des résultats en arborescence que l'on verra en 5D.

Remarque 3 - Le temps que la machine perd à faire ces recherches est largement récupéré par la suite. Chaque résultat que l'on aurait archivé à tort aurait d'abord pris de la place, mais surtout aurait conduit à faire des essais qui auraient donné d'autres résultats inintéressants à ce sens et que l'on aurait encore gardés. L'élimination de résultats est toujours payante. On peut considérer qu'en moyenne ces recherches suppriment la moitié des résultats dont l'intérêt calculé aurait justifié un archivage. Le cas le plus fréquent est, de loin, le premier.

Remarque 4 - On a employé une élimination heuristique de résultats qui tient compte de ce que l'on a déjà trouvé, mais est assez différente de la méthode précédente. Elle s'est imposée par suite du grand nombre de productions de type substitutif qui existent dans certaines axiomatique; d'où prolifération des MMthéorèmes qui s'en déduisent. Pour y remédier, on a défini une mesure de la complexité d'une suite de symboles et on n'a gardé que les MMthéorèmes $A \longrightarrow B \Rightarrow C$ qui avaient la complexité la plus faible pour A.

Ceci n'a joué qu'un rôle épisodique et aurait mérité d'être passé sous silence. Mais on peut y voir le germe d'une heuristique très puissante qui peut s'adapter à une grande variété de problèmes : on définit un type de résultats et une mesure de complexité pour ces résultats. On ne garde que ceux dont la complexité est inférieure à un seuil dont la position se calcule en fonction du résultat qui a la complexité la plus faible. On peut ainsi traiter des axiomatiques de difficultés très différentes, la sévérité des heuristiques s'adaptant à la difficulté de l'axiomatique même si cette difficulté évolue en cours d'étude. Le programme n'a pas utilisé assez systématiquement des heuristiques de ce type.

d - Conclusion

Cette élimination qui tient compte des autres résultats que l'on a trouvé complète l'élimination des résultats dont l'intérêt est trop faible. Le programme passe la plus grande partie de son temps à faire ces recherches. On a essayé de l'accélérer en archivant les résultats en arborescence : on peut trouver tout de suite ce que l'on cherche ou mener de front plusieurs recherches.

Un seul sous programme fait toutes les recherches dont nous venons de parler. Une recherche particulière est précisée par 7 constantes qui sont lues. Cela donne une très grande souplesse et on a effectivement été amené à considérer des éliminations qui n'avaient pas été prévues initialement. On avait à la fin 30 types de recherches simples et 10 types de recherche à 2 niveaux (comme en c). A chaque type de résultat (théorème, production à 1 ou 2 antécédents, MMthéorème) correspond un certain nombre de recherches qui peuvent, soit éliminer des résultats déjà archivés, soit amener à considérer le résultat étudié comme sans intérêt.

Par suite de la longueur de ces recherches, on ne les fait que pour les résultats dont l'intérêt est tel qu'ils devraient être archivés. Bien que de toutes les façons on ne les archive pas, il aurait pu être intéressant d'étudier les autres, car dans certains cas un résultat d'intérêt faible peut entrainer l'élimination d'un résultat intéressant. Comme c'est assez rare, on préfère prendre le risque de ne pas éliminer de temps en temps un résultat qui devrait l'être et ne pas perdre un temps considérable.

3C - ELIMINATION D'ESSAIS

a - Introduction

Quand on veut essayer un résultat, on a intérêt à ne pas l'essayer avec tous les résultats que l'on a trouvé. Certains essais ont très peu de chance de conduire à un résultat intéressant et il vaut mieux ne pas les faire. L'analyse de l'intérêt du résultat les éliminerait, mais elle prend un temps important. Cette élimination n'a pas pour but de gagner de la place en mémoire, car les résultats inutiles sont supprimés quand on étudie leur intérêt. Elle a pour but de gagner du temps.

On l'utilise uniquement quand on applique une production à un théorème.

b - Définition de l'intérêt d'un essai

C'est la somme de l'intérêt des deux résultats essayés et de l'intérêt couplé que l'on va définir.

L'intérêt couplé va essayer d'estimer "l'accord" qu'il y a entre les deux résultats.

- Si la production comporte des parties neutres ou a une variable qui figure dans l'antécédent et pas dans le résultat (comme q dans ↄↄpqr⇒ↄ~pr), on compte le nombre de connectives unaires et binaires, soient u et b, qui figurent dans les termes que l'on va substituer à ces variables pour faire coïncider l'antécédent avec le théorème. On ajoute alors à l'intérêt couplé Au+Bb, A et B étant des données. Il est normal de tenir compte de ce que des parties importantes du théorème ne seront pas modifiées ou seront supprimées.

- On va compter le nombre de connectives unaires et binaires qui figurent dans les termes substitués aux variables du théorème ou de l'antécédent au cours de l'opération qui les amène à coïncider.

Exemple 1 - ↄ~~pp+ↄ~P~Q⇒ↄQP⊣↦?
à la variable p on substitue ~Q
à la variable P on substitue ~p

D'où pour le théorème : $u_1=1$; $b_1=0$
pour l'antécédent: $u_2=1$; $b_2=0$

Exemple 2 - ↄpↄqp+ↄↄPQR⇒ↄQR⊣↦?
à p on substitue ↄPQ

à R on substitue ⊃qp

D'où : $u_1=0$; $b_1=1$

$u_2=0$; $b_2=1$

L'expérience montre qu'il est préférable que les substitutions se localisent soit dans le théorème, soit dans l'antécédent. Pour tenir compte de cela, on diminue l'intérêt couplé de la plus petite des deux valeurs : Cu_1+Db_1 et Cu_2+Db_2, C et D étant des constantes positives.

c - *Conclusion*

On ne fera pas tout essai dont l'intérêt est inférieur à un certain seuil. On voit que si un résultat est très intéressant, on sera plus libéral pour faire des essais avec lui, car l'intérêt des essais sera en général grand puisque son intérêt y figure. L'intérêt couplé essaye de favoriser les essais de couples production-theorème qui sont bien adaptés.

En réalité cette élimination joue un rôle assez secondaire. Le problème de la sélection des essais n'a pas été résolu de façon satisfaisante. Ce n'est pas très grave, car on a derrière un bon crible qui éliminera les résultats sans intérêt, mais cela augmente le temps machine. On ne perd pas trop de temps à faire les dérivations, car on les fait en grande partie en parallèle. Les heuristiques d'élimination des essais en éliminent environ 25 %, ce chiffre variant largement d'une axiomatique à l'autre. Nous verrons au moment de l'ambiguité, qu'une autre méthode contribue à éliminer des essais (cf 4B).

Remarquons que la méthode utilisée est très différente de celle du mathématicien qui sélectionne très sévèrement les essais qu'il va faire. Son "intuition" lui indique avec quel théorème il va être fructueux d'essayer telle production. Le mathématicien ne peut prendre la méthode utilisée par le programme : il perdrait trop de temps et ferait trop d'erreurs. La méthode utilisée est bien adaptée à l'ensemble du programme, c'est pourquoi nous l'avons prise : notre but n'est pas de simuler l'intelligence humaine, mais de faire de l'intelligence artificielle.

3D - INFLUENCE DE L'ORDRE DANS LEQUEL ON FAIT LES ESSAIS

Ceci a une très grande importance. L'expérience montre que les essais les plus fructueux sont ceux où l'un au moins des deux résultats qui y participe a un intérêt très élevé. Il ne peut être question de prouver ceci, qui n'est qu'une loi de nature heuristique. Il serait facile de trouver un exemple de deux résultats peu intéressants qui donnent un résultat intéressant, mais en général c'est faux. Cette loi est du reste en parfait accord avec le bon sens. Si l'on disposait d'un temps et d'une capacité de mémoire illimités, l'ordre dans lequel on fait les essais n'aurait pas la moindre importance. Mais dans un cas pratique il est capital d'avoir rapidement les résultats les plus intéressants. Les premiers programmes avaient pour politique d'essayer les résultats dans l'ordre dans lequel on les trouvait. La rapidité avec laquelle on trouvait les théorèmes importants a énormément augmenté quand on a décidé d'essayer d'abord les résultats ayant l'intérêt le plus élevé.

Essayer les résultats dans l'ordre où on les a trouvés revient à décrire le graphe vu en 3A en épuisant entièrement un niveau avant de passer au suivant. Vu la taille du graphe, même élagué par les heuristiques, cela interdirait d'atteindre un grande profondeur. Quand on choisit au contraire le résultat le plus intéressant, s'il est déjà à une grande profondeur, on aura des résultats encore plus profonds, que l'on essayera du reste immédiatement s'ils sont, eux aussi, très intéressants.

Les deux arborescences ci-dessus demandent le même temps machine, mais l'une va au maximum à la profondeur 3 alors que l'autre va jusqu'à 8. On a effectivement obtenu pour des résultats importants des démonstrations dont la profondeur dépassait 30. Cela était impensable avec l'autre méthode.

3E - INTERET DE L'UTILISATION DES DIVERS NIVEAUX LINGUISTIQUES

Toutes les dérivations que le programme est capable de faire pourraient s'obtenir à partir de deux MMthéorèmes :

$$a;a \Rightarrow b \longrightarrow b$$
$$a;b;a,b \Rightarrow c \longrightarrow c$$

qui ne font que formaliser la manière dont on doit utiliser les productions à un et deux antécédents de l'axiomatique. Or nous avons considéré bien d'autres MMthéorèmes et même des MMMthéorèmes qui sont énoncés en 2D. Nous l'avons fait uniquement pour des raisons heuristiques : on n'espère pas donner au programme la possibilité de démontrer plus de théorèmes. On veut simplement lui donner plus de facilité pour reconnaitre quelles sont les dérivations qu'il est intéressant de faire.

D'une manière générale, au lieu d'appliquer :

$$a;b;a,b \Rightarrow c \longrightarrow c$$

nous avons appliqué :

$$a,b \Rightarrow c \rightsquigarrow a \longrightarrow b \Rightarrow c$$
$$a \Box a \longrightarrow b \Rightarrow c \rightsquigarrow b \Rightarrow c$$
$$b;b \Rightarrow c \longrightarrow c$$

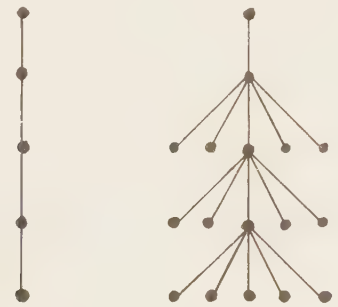
Le résultat est évidemment le même, mais la façon de l'obtenir est différente. Nous allons voir les trois raisons importantes de cette décomposition à première vue superflue.

1 - Si l'on veut appliquer a;b;a,b⇒c⟶ c on doit définir simultanement les théorèmes que l'on va faire coïncider avec a et b et la production qui viendra à la place de a, b⇒c. Il faut donc choisir trois résultats ce qui risque de mener à beaucoup de combinaisons. Il est plus commode de ne choisir que 2 résultats à chaque étape. Cette objection est toute théorique, car dans la pratique il n'y a en général qu'une seule production à 2 antécédents.

On voit aussi que l'on ne distingue pas la dissymétrie des antécédents de la production. a et b jouent un rôle équivalent. Dans la méthode choisie a,b⇒c⟿a⟶b⇒c leur fait jouer un rôle très différent. C'est important en logique où les antécédents du modus ponens ont des formes distinctes.

Enfin, on a d'autres possibilités que la décomposition en trois étapes donnée plus haut. En particulier on peut utiliser la transitivité de ⇒. Créer des Mthéorèmes nouveaux que l'on peut combiner entre eux donne un grand nombre de possibilités qui ne paraissent pas évidentes si l'on se contente d'utiliser la forme a;b;a,b⇒c. Le paragraphe suivant éclairera ce point.

2 - Cela permet d'éviter de considérer des intermédiaires sans intérêt. Un Mthéorème est un raccourci évalué une fois pour toutes qui fait sauter plusieurs étapes. L'utiliser revient à sauter plusieurs niveaux dans le graphe combinatoire représentant toutes les démonstrations possibles.



Cela revient à avoir une figure linéaire comme celle de gaûche au lieu de considérer une arborescence importante comme celle de droite. Prenons un exemple pour éclaircir cela : dans l'axiomatique 1 une formule de démonstration de : ⊃p⊃qr⇒⊃q⊃pr est RbRRRiaba. Il est naturel de l'appliquer à ⊃⊃pq⊃⊃rp⊃rq de formule d et on en déduit le nouveau théorème : ⊃⊃pq⊃⊃qr⊃pr de formule RbRRRdaba.

Il aurait fallu autrement des dérivations :

⊃⊃pq⊃⊃rp⊃rq : d ⊃⊃p⊃qr⊃⊃pq⊃pr:a

⊃⊃⊃pq⊃rp⊃⊃pq⊃rq:Rda

⊃p⊃qp:b

⊃s⊃⊃⊃pq⊃rp⊃⊃pq⊃rq:RRdab

⊃⊃p⊃qr⊃⊃pq⊃pr:a

⊃⊃s⊃⊃pq⊃rp⊃s⊃⊃pq⊃rq:RRRdaba

⊃p⊃qp:b

⊃⊃rp⊃⊃pq⊃rq:RbRRRdaba

Il aurait été nécessaire de passer par trois résultats sans intérêt. On n'aurait pas trouvé forcément quel était le seul résultat avec lequel il fallait les essayer. Peut être même on n'aurait pas vu qu'il fallait les garder pour faire des essais.

Le Mthéorème ⊃p⊃qr⇒⊃q⊃pr peut s'appliquer à bien d'autres théorèmes :

avec ⊃pp il donne ⊃p⊃⊃pqq
avec ⊃~p⊃pq il donne ⊃p⊃~pq
avec ⊃⊃p⊃qr⊃⊃pq⊃pr il donne ⊃⊃pq⊃⊃p⊃qr⊃pr

et dans tous les cas il aurait fallu démontrer trois intermédiaires inutiles qui auraient conduit soit à l'abandon de cette voie, soit à une multiplication des résultats et des essais.

On pourrait objecter qu'il suffirait d'attendre que l'on ait démontré ⊃⊃p⊃qr⊃q⊃pr. Or, d'une part ⊃p⊃qr⇒⊃q⊃pr peut se démontrer bien avant et d'autre part il arrive très souvent que le théorème dont on peut déduire le Mthéorème par un MMthéorème (comme ⊃pq⟶p⇒q) ne se démontre que bien après ce Mthéorème qui peut même être indispensable pour sa démonstration. L'étude de l'axiomatique 2 est très instructive à ce point de vue.

3 - Pour évaluer l'intérêt des théorèmes, il est capital de leur appliquer des MMthéorèmes pour voir les Mthéorèmes qui s'en déduisent. Il est très facile de donner des règles générales pour évaluer l'intérêt d'une production à un antécédent : il suffit de comparer les deux suites de symboles antécédent et résultat. On est assez embarassé pour affecter un intérêt à la suite de symboles qui constitue le théorème. Le nombre de connectives est un critère grossier : ⊃⊃pq⊃⊃rp⊃rq et ⊃p⊃q⊃r⊃s⊃ts ont le même nombre de connectives !

Il est bien plus intéressant d'appliquer une fois pour toutes a□a⟶b⇒c⇝b⇒c et de répercuter sur a l'intérêt trouvé pour b⇒c. Cela correspond d'ailleurs un peu à la façon dont nous opérons nous même. Comme nous l'avons déjà dit, cette manière d'agir est liée au fait que l'on a travaillé sur des axiomatiques de la logique.

Nous venons de voir trois raisons qui justifient l'approche que l'on a prise. La troisième est liée au fait que l'on étudie des axiomatiques logiques, la première perd de son intérêt s'il n'y a qu'une seule production. La plus importante est la seconde qui permet de détecter des schémas de suites d'opération qui reviennent souvent. Les Mthéorèmes sont du reste fréquemment utilisés par les logiciens. (cf exemple les règles dans Hilbert et Ackermann - Mathematical logic).

CHAPITRE IV

QUELQUES PROBLEMES ANNEXES

4A - LES DEFINITIONS

4A1 - Généralités

Une définition indique que deux suites de symboles α et β doivent être considérées comme identiques. On l'écrit $\alpha=\beta$. Si $\gamma\alpha\delta$ est un théorème, $\gamma\beta\delta$ en est également un et inversement. Si la suite α apparait plusieurs fois dans un énoncé, on peut remplacer α par β pour certaines occurrences et ne pas le faire pour d'autres. On voit la différence avec les substitutions habituelles : si on substitue un terme à une variable, on doit le faire pour toutes les occurrences de la variable. Remarquons qu'il faut distinguer les Mconnectives "=" et " $\Leftrightarrow$ " qui est l'équivalence de la Mlangue $A \Leftrightarrow B$ est une abréviation de $A\Rightarrow B, B\Rightarrow A$. Si l'on a $A \Leftrightarrow B$ on ne peut en déduire que $\gamma A \delta$ entraine $\gamma B \delta$ et inversement que si γ et δ se réduisent à la suite vide.

On introduit en général les définitions dans un but abréviatif : une certaine suite de symboles se rencontre souvent et on crée un nouveau symbole pour la répresenter et avoir ainsi des théorèmes plus compacts. On peut aussi être amené à introduire une définition pour représenter uhe suite de symboles qui a un sens très intéressant dans une interprétation particulière du système. Une des suites de symboles soit α a une longueur inférieure à celle de β et l'on a tendance à appliquer la définition dans le sens : substituer α à β. Dans certains cas il peut être utile de revenir à la suite de départ, en particulier pour éviter un échec au cours d'une abréviation.

Exemples : $\wedge pq=\sim\supset p\sim q$

$\supset p\supset q\sim\supset p\sim q$ devient $\supset p\supset q\wedge pq$

$\supset\sim\supset p\sim qq$ devient $\supset\wedge pqq$

$\equiv pq=\wedge\supset pq\supset qp$

$\wedge\supset pp\supset pp$ devient $\equiv pp$

$\supset\wedge\supset pq\supset qp\supset pq$ devient $\supset\equiv pq\supset pq$

Dans certains cas, il peut être nécessaire de faire des substitutions dans un résultat pour pouvoir lui appliquer une définition. Si on applique la définition en sens inverse, on aura un résultat plus faible que celui dont on est parti :

Exemples : 1 - $\supset p\supset q\wedge pq$. Par la substitution $p/\supset pq; q/\supset qp$, il devient : $\supset\supset pq\supset\supset qp\wedge\supset pq\supset qp$ soit $\supset\supset pq\supset\supset qp\equiv pq$.

2 - $\supset\sim\supset pqp$. Par la substitution $p/p; q/\sim q$, il devient $\supset\sim\supset p\sim qp$ soit $\supset\wedge pqp$.

3 - $\supset\wedge pqp$. On effectue la substitution : $p/\supset pq; q/\supset qp$ et l'on a : $\supset\wedge\supset pq\supset qp\supset pq$ soit $\supset\equiv pq\supset pq$

4 - On a la définition $\supset pq=\vee\sim pq$ et le théorème :

$\supset\supset pq\supset\vee rp\vee rq$. On substitue $\sim r$ à r : $\supset\supset pq\supset\vee\sim rp\vee\sim rq$ soit $\supset\supset pq\supset\supset rp\supset rq$
(ou $\supset\supset pq\supset\vee\sim rp\supset rq$ ou $\supset\supset pq\supset\supset rp\vee\sim rq$).

Le programme considère comme distincts deux résultats qui se déduisent l'un de l'autre par application d'une définition. S'ils sont tous deux intéressants, on les archive tous les deux. On peut très bien n'en archiver qu'un ou aucun. N'archiver qu'une des formes aurait compliqué le programme de dérivation en l'obligeant à transformer les suites de symboles pour éviter certains échecs dûs à la différence des connectives. Si on a les deux formes en mémoire, l'une pourra donner échec, mais l'autre donnera le résultat cherché.

Exemple : ⊃Λpqq+⊃~pq⇒⊃~qp ⟶ Echec

⊃~⊃p~qq+⊃~pq⊃~~qp ⟶ ⊃~p⊃q~p

4A2 - Application d'une définition

On va appliquer "localement" la méthode de dérivation définie en 2B. Il faut par des substitutions amener à coïncider une partie d'un énoncé avec un des membres de la définition. On remplace alors la partie de l'énoncé qui coïncide par l'autre membre. Les substitutions peuvent avoir lieu aussi bien dans le résultat que dans la définition. On applique donc la méthode générale en mettant sur la première ligne le terme du théorème qui commence à l'endroit où l'on veut faire la substitution, sur la deuxième ligne le membre de la définition que l'on veut faire coïncider et sur la troisième ligne le théorème où l'on a remplacé le terme mis sur la première ligne par l'autre membre de la définition. Bien entendu, les variables dans le théorème et dans la définition doivent être différentes.

Exemple : 1 - On veut appliquer ΛPQ=~⊃P~Q au terme qui commence au deuxième symbole de ⊃~⊃pqp

1 - ~⊃pq
~⊃P~Q
⊃ΛPQp

2 - ⊃pq
⊃P~Q
⊃ΛPQp

3 - pq
P~Q
⊃ΛPQp

4 - q
~Q
⊃ΛPQP

5 - Λ
Λ
⊃ΛPQP

d'où le résultat ⊃Λpqp (qui est plus faible que ⊃~⊃pqp : appliquée en sens inverse la définition donne ⊃~⊃p~qp)

Exemple : 2 - On veut appliquer ≡PQ=Λ⊃PQ⊃QP au terme qui commence au deuxième symbole de

⊃~⊃pq⊃q⊃pr⊃~q~p

1 - Λ⊃pq⊃q⊃pr
\⊃PQ⊃QP
⊃≡PQ⊃~q~p

2 - ⊃pq⊃q⊃pr
⊃PQ⊃QP
⊃≡PQ⊃~q~p

3 - pq⊃q⊃pr
PQ⊃QP
⊃≡PQ⊃~q~p

4 - q⊃q⊃Pr
Q⊃QP
⊃≡PQ⊃~q~P

5 - ⊃Q⊃Pr
⊃QP
⊃≡PQ⊃~Q~P

6 - Q⊃Pr
QP
⊃≡PQ⊃~Q~P

7 - ⊃Pr
P
⊃≡PQ⊃~Q~P

et il y a échec car P figure dans le terme ⊃Pr avec lequel il devrait être amené à coïncider.

Les deux causes d'échec étudiées en 2B subsistent : les connectives sont différentes ou l'on doit faire coïncider une variable avec un terme qui la contient.

4A3 - Heuristiques utilisées pour appliquer les définitions

Nous venons de voir comment on applique les définitions. Il reste à préciser quand on les applique. Sans cela on risquerait de faire beaucoup trop d'essais. Comme nous l'avons dit un des

membres soit α est plus long que l'autre. Cela définit un sens d'application : on essayera de remplacer α par β. Deux cas sont à distinguer.

1 - α commence par deux connectives. Si dans la suite de symboles qu'est l'énoncé du théorème, on rencontre ces deux connectives dans le même ordre, on essaye systématiquement d'appliquer la définition.

Exemple : si l'on a $\wedge C pq C qp = \equiv pq$, on essayera d'appliquer la définition à tous les termes figurant à l'intérieur d'un théorème et commençant par "$\wedge C$".

Ceci ne conduit pas à faire trop d'essais : l'expérience montre que la combinaison de connectives correspondant à un début de définition se rencontre rarement et quand on le rencontre l'application de la définition a, en général, un heureux résultat.

D'autre part, la plupart des définitions ont leur partie la plus longue qui commence par deux connectives (en notation polonaise). Une exception : $|pp = \sim p$.

2 - On a rencontré dans l'énoncé un terme qui commence par la première connective de α, mais cette connective est suivie par une variable. En substituant un terme judicieusement choisi à cette variable, on peut espérer pouvoir appliquer la définition. Mais d'une part cette substitution risque d'allonger le théorème si la variable a plusieurs occurrences, ce qui va contre le but cherché. D'autre part, ce cas se rencontre beaucoup trop fréquemment (et même plusieurs fois à l'intérieur du même théorème). Pour limiter le nombre des essais, on ne les fera que pour les résultats dont l'intérêt est supérieur à un certain seuil.

Exemple : $C \wedge pqp$ étant très intéressant, on essayera de lui appliquer $\wedge C pq C qp = \equiv pq$ ce qui donnera $C \equiv pq C qp$ bien qu'il n'y ait coïncidence que du $\wedge$.

Ce cas est le seul où l'on puisse appliquer les définitions qui commencent par une seule connective.

4A4 - Conclusion

Cette méthode très empirique donne des résultats à peu près acceptables dans certaines axiomatiques. Elle se prête mal à des définitions du type $|pp = \sim p$ (qui sont heureusement rares).

La principale faiblesse du programme n'est pas dans l'application : quand un résultat mérite l'application d'une définition, le programme le fait en général. Mais il ne provoque pas la découverte de tels résultats. Il devrait chercher à obtenir les suites de connectives que la définition abrégera.

Exemple : $CCp \sim q C q \sim p + C pq \Rightarrow C \sim q \sim p \leftrightarrow C \sim C p \sim q \sim C q \sim p$ qui après une double application de la définition donnera $C \wedge pq \wedge qp$.

Le programme appliquerait certainement la bonne définition à $C \sim C p \sim q \sim C q \sim p$ s'il l'obtenait, mais il ne l'obtient pas parcequ'il n'applique pas $C pq \Rightarrow C \sim q \sim p$, qui est une production faible, aux théorèmes. Cela va contre ses heuristiques. Ses heuristiques sont trop grossières pour se rendre compte que l'application de cette production est utile en liaison avec la définition du $\wedge$, car elle a des chances de faire apparaitre le groupement $\sim C$. Il y a là une lacune du programme. Il est difficile de trouver une méthode simple et générale pour y remédier.

4B - L'AMBIGUITE

4B1 - Généralités

Dans les systèmes formels utilisés dans l'étude des langages, on essaye d'éviter l' ambiguité, c'est-à dire la possibilité d'obtenir la même suite de symboles à partir de dérivations différentes. En effet le sens que l'on affecte à un énoncé dépend en partie de la façon dont on peut l'obtenir, c'est à dire de sa formule de démonstration. Il faut éviter de donner des sens différents au même énoncé.

Dans les systèmes formels utilisés en mathématiques, on ne s'intéresse en général qu'aux suites de symboles que l'on peut dériver, et non à leur démonstration. L'ambiguité n'est pas génante, aussi la plupart de ces systèmes sont très ambigüs. Nous verrons même qu'il y a des systèmes où chaque théorème possède une infinité de démonstrations. Cela impose tout de même au programme de vérifier, chaque fois qu'il a un résultat nouveau, s'il n'a pas déjà obtenu cet énoncé. Nous avons vu en 3B3 que cette recherche existait.

Il faut distinguer deux sortes d'ambiguité. L'une n'est qu'apparente et vient de l'utilisation des Mthéorèmes : on peut obtenir de diverses façons la même formule de démonstration. L'autre correspond vraiment à de l'ambiguité : au même énoncé peuvent correspondre des formules de démonstration différentes.

4B2 - Ambiguité dûe à l'utilisation des Mthéorèmes

Ce n'est pas une véritable ambiguité. Certes des dérivations différentes conduisent au même énoncé, mais toutes ces dérivations correspondent à une seule formule de démonstration. Nous avons vu en 2C1 qu'à une formule de démonstration contenant n fois le nom R d'une production à 2 antécédents, on pouvait faire correspondre 2n décompositions différentes du type : A+B⇒C↣D, D étant le théorème correspondant à la formule.

C'est ainsi qu'à la formule :

RaRRRRbbRRbRRabaaba on peut faire correspondre 18 dérivations. Certaines sont simples comme : ⊃⊃p⊃qr⊃⊃pq⊃pr+⊃p⊃⊃qrs⇒⊃p⊃rs↣⊃⊃p⊃qr⊃q⊃pr

d'autres extrèmement compliquées, comme :

⊃p⊃qp+⊃⊃⊃p⊃qr⊃⊃pq⊃pr⊃⊃⊃st⊃su⊃⊃v⊃⊃st⊃su⊃⊃w⊃ xwy⇒ ⊃⊃s⊃tuy↣⊃⊃p⊃qr⊃q⊃pr

Heureusement, en général la plupart de ces 2n décompositions sont très complexes et n'ont pas la moindre chance d'être considérées par le programme. Il est bien évident que personne, mathématicien ou programme, ne pourrait avoir l'idée de faire la dernière dérivation donnée en exemple. Il peut toutefois arriver qu'il y en ait deux ou trois d'intuitives que le programme peut être amené à essayer. Bien sûr, il verra quand il aura obtenu des résultats qu'ils ne sont pas nouveaux, mais on a préféré avoir un dispositif qui écarte ces essais à priori dans la plupart des cas où cela risque de se produire.

Il y a une associativité de l'opération + que l'on peut écrire :

(A+B)+C = A+(B+C) ce qui signifie :

si A+B ↣ D
D+C ↣ E
B+C ↣ F
B+C ↣ F
alors A+F ↣ E

A, D, E étant des théorèmes et B, C, F des productions à un antécédent. Dans les deux cas on a

en effet les mêmes formules de démonstration :

si A a pour formule α;B:$\beta i \gamma$;C:$\delta i \theta$
D a pour formule : $\beta \alpha \gamma$
E résultant de D+C : $\delta \beta \alpha \gamma \theta$
F a pour formule : $\delta \beta i \gamma \theta$

et le résultat de A+F a bien pour formule : $\delta \beta \alpha \gamma \theta$

Exemple : A est $\supset p \supset \sim pq$
B est $\supset p \supset qr \Rightarrow \supset \supset pq \supset pr$
C est $\supset \supset pqr \Rightarrow \supset qr$
D est $\supset \supset p \sim p \supset pq$
F est $\supset p \supset qr \Rightarrow \supset q \supset pr$
E est $\supset \sim p \supset pq$

La méthode que l'on va décrire oblige d'archiver toutes les combinaisons qui ont donné naissance à un résultat et tous les essais que l'on a fait avec ce résultat.

Supposons que l'on veuille appliquer la production F au théorème A

1 - Si F peut être obtenu par action de la production C sur la production B : B+C$\rightarrowtail$F, on regarde si l'on n'a pas déjà essayé la production B sur le théorème A. Si oui, on ne fait l'essai que si l'on n'a pas encore appliqué la production C au théorème qui en a résulté : A+B.

2 - Si A peut être obtenu par action de la production H sur le théorème G : G+H$\rightarrowtail$A, on regarde si l'on n'a pas déjà essayé la production F sur la production H donnant un résultat J : H+F$\rightarrowtail$J. Si oui, on ne fait l'essai que si on n'a pas encore appliqué la production J au théorème A.

Cette méthode est relativement facile à programmer, mais augmente la quantité d'information mémorisée. Elle élimine la plus grande partie des "fausses" ambiguités, dues à l'utilisation des Mthéorèmes. On aurait pu prévoir des cas plus complexes, mais cela aurait alourdi beaucoup le programme pour des cas se rencontrant rarement. Les conséquences n'en sont pas très graves, puisque le programme s'aperçoit tout de suite après la dérivation qu'il a déjà obtenu le résultat. Cette perte de temps est admissible si elle n'est pas trop fréquente.

4B3 - Ambiguité effective

Au même énoncé correspondent plusieurs formules de démonstration. On a déjà vu en 2C2 qu'un théorème comme $\supset \supset p \supset qr \supset q \supset pr$ avait plusieurs démonstrations différentes dans l'axiomatique 1 et l'on a alors donné les formules de démonstration qui leur correspondait. Ce phénomène de l'ambiguité apparait tout à fait normal au mathématicien habitué à avoir une grande diversité de possibilités pour démontrer ce qu'il cherche. On citait le cas d'un professeur de taupe qui avait collectionné une cinquantaine de démonstrations du théorème de Menelaüs.

Il est difficile d'analyser les causes profondes de l'ambiguité. Certains aspects sont toutefois évidents :

1 - On a une production B$\Rightarrow$C qui est un élément neutre à droite pour l'opération + :

A+B$\Rightarrow$C$\rightarrowtail$A

C'est le cas où B$\Rightarrow$C est la production : p$\Rightarrow$p. Si sa formule de démonstration est $\alpha i \beta$ (α et β ne se réduisant pas tous deux à la suite vide), si un théorème a pour formule de démonstration γ, il en

aura une infinité données par : $\alpha^n\gamma\beta^n$. (n= 1, 2,...). On voit donc qu'il existe des axiomatiques où chaque théorème a une infinité de démonstrations. Ce cas n'est pas exceptionnel : en logique des propositions, on a très souvent cette production (obtenue par $\supset pp$ et $\supset pq \longrightarrow p \Rightarrow q$ ou $|p|pp$ et $|p|qr \longrightarrow p \Rightarrow r$ dans l'axiomatique 5).

Il se peut aussi que la production ne soit que partiellement neutre : ou elle redonne le résultat initial, ou il y a échec.

Exemple : $\supset pq \Rightarrow \supset pq$. Tout théorème commençant par $\supset$ sera obtenu à nouveau. Il y aura échec pour tous les théorèmes commençant par une autre connective (par exemple $\sim\wedge p \sim p$). Ce cas est très fréquent, car souvent l'inverse d'une production est une production : $A \Rightarrow B$ et $B \Rightarrow A$ sont des productions (si c'est la même production, elle est autoréciproque comme on l'a vu en 3B1d). Si on les applique successivement on a les productions $A \Rightarrow A$ ou $B \Rightarrow B$.

Exemples : $\supset p \supset qr \Rightarrow \supset q \supset pr + \supset p \supset qr \Rightarrow \supset q \supset pr \longmapsto \supset p \supset qr \Rightarrow \supset p \supset qr$

$\supset \sim p \sim q \Rightarrow \supset qp + \supset pq \Rightarrow \supset \sim q \sim p \longmapsto \supset \sim p \sim q \Rightarrow \supset \sim p \sim q$

Les ambiguités de ce type sont évitées de 2 façons :

- le programme élimine toutes les productions où un antécédent est identique au résultat

Exemple : $p, \supset pq \Rightarrow \supset pq$

- pour toute production nouvelle, il regarde s'il existe la production inverse (qui peut être la production elle même). Il veille ensuite à ne pas appliquer la production $B \Rightarrow A$ au théorème C obtenu par action de $A \Rightarrow B$ sur un autre théorème. On retrouverait en effet le théorème de départ ou on aurait échec.

Remarquons que si $\alpha i \beta$ est une formule de démonstration d'une production autoréciproque, $\alpha^{2n} i \beta^{2n}$ est la formule de démonstration d'une production partiellement neutre du type $A \Rightarrow A$. D'où une nouvelle source d'une infinité de démonstrations, mais pour un sous ensemble des théorèmes : ceux à qui on peut appliquer la production $A \Rightarrow A$.

2 - Il arrive très fréquemment que l'application d'une production, bien que n'étant pas neutre, à un théorème redonne ce théorème.

Exemples : $\supset p \supset q \supset rq + \supset\supset pqr \Rightarrow \supset qr \longmapsto \supset p \supset q \supset rq$

$\supset p \supset qq + \supset\supset pqr \Rightarrow \supset qr \longmapsto \supset p \supset qq$

Si la production a pour formule $\alpha i \beta$ et le théorème γ, ce dernier aura aussi l'infinité de formules $\alpha^n\gamma\beta^n$. En réalité on applique $\supset\supset pqr \Rightarrow \supset qr$ à $\supset\supset ps \supset q \supset rq$ qui se déduit par la substitution : $p/\supset ps$ du théorème initial. Ce phénomène se produit avec les productions fortes au sens de la première définition : une substitution va affaiblir le résultat et la production le ramènera au niveau initial.

Plus généralement tout résultat que l'on retrouve après lui avoir appliqué un nombre quelconque de productions a une infinité de démonstrations. Si au départ il avait la formule γ et quand on le retrouve il a une formule du type $\alpha\gamma\beta$, il aura aussi l'ensemble de formules $\alpha^n\gamma\beta^n$. Ce cas se produit assez souvent :

$\supset p \sim \sim p + \sim p \sim q \Rightarrow \supset qp \supset \sim \sim pp$

$\supset \sim \sim pp + \supset \sim p \sim q \Rightarrow \supset qp \longmapsto \supset p \sim \sim p$

Si $\supset \sim p \sim q \Rightarrow \supset qp$ a pour formule $\alpha i \beta$ et $\supset p \sim \sim p : \gamma$, on voit que $\supset \sim \sim pp$ aura les formules $\alpha^{2n+1}\gamma\beta^{2n+1}$ et $\supset p \sim \sim p$ les formules $\alpha^{2n}\gamma\beta^{2n}$.

Le phénomène peut être plus complexe encore, comme le montre un exemple dû à Tarski :

Soit $\supset\supset p \supset qr \supset q \supset pr : d ; \supset p \supset\supset q \supset pr \supset qr : e ; p, \supset pq \Rightarrow q : Rij$

on a ⊃p⊃⊃q⊃pr⊃qr : Rdd

⊃⊃p⊃qr⊃q⊃pr : ReRee (en effet Ree est : ⊃⊃p⊃⊃q⊃⊃r⊃qs⊃rst⊃pt)

Donc si d est une formule de démonstration du premier théorème RRddRRddRdd en est une autre et si e en est une du second, RReReeReRee en est une autre. On a encore une infinité de formules pour chacun des deux énoncés. Remarquons que les substitutions peuvent se faire indépendamment : on n'est pas obligé de faire la même substitution pour toutes les occurrences de d(ou de e). Cela conduit à un nombre énorme de formules possibles.

Des phénomènes analogues ont lieu quand on fait agir une production sur une autre production :

p⇒⊃qp + ⊃⊃pqr⇒⊃qr ↔ p⇒⊃qp

~p⇒⊃pq + p⊃pq⇒⊃pq ↔ ~p⇒⊃pq

Ce type d'ambiguité est très fréquent, mais n'est pas génant, car on n'archive un énoncé qu'une seule fois. Si on le retrouve, on ne l'archivera pas et on éliminera du même coup l'infinité de démonstrations qui pourraient correspondre à cet énoncé.

3 - Il arrive souvent qu'il y a commutativité dans l'ordre d'application de deux ou plusieurs productions.

Si A⇒B + C⇒D ↔ E⇒F

C⇒D + A⇒B ↔ G⇒H

il n'y a aucune raison pour que les productions E⇒F et G⇒H soient les mêmes (Dans le premier cas on fait coïncider B et C et dans le second D et A).

Il se trouve assez souvent qu'elles soient identiques. Mais elles ont deux formules de démonstration distinctes : si A⇒B a pour formule αiβ et C⇒D : γiδ alors E⇒F aura pour formule γαiβδ et G⇒H : αγiδβ.

Exemple : ⊃⊃pqr⇒⊃qr + ⊃p⊃⊃q⊃rqs⇒⊃ps ↔ ⊃⊃pq⊃⊃r⊃srt⇒⊃qt

⊃p⊃⊃q⊃rqs⇒⊃ps + ⊃⊃pqr⇒⊃qr ↔ ⊃⊃pq⊃⊃r⊃srt⇒⊃qt

Il y a même des cas plus remarquables. Soient les 3 productions :

⊃⊃pqr⇒⊃qr de formule Ai

⊃⊃⊃pq⊃prs⇒⊃⊃p⊃qrs de formule Bi

⊃p⊃⊃q⊃rqs⇒⊃ps de formule Ci

Aux formules : ABCi ; CABi ; ACBi correspondent l'énoncé : ⊃⊃p⊃⊃qr⊃qs⊃⊃t⊃utv⇒⊃⊃q⊃rsv
et aux formules BACi ; BCAi ; CBAi correspondent : ⊃⊃⊃pq⊃pr⊃⊃s⊃tsu⇒⊃⊃qru
(on avait déjà vu que AC et CA correspondaient à la même production).

Ceci va donner trois démonstration de ⊃⊃p⊃qr⊃q⊃pr :

⊃⊃p⊃qr⊃⊃pq⊃pr + ⊃⊃p⊃⊃qr⊃qs⊃⊃t⊃utv - ⊃⊃q⊃rsv ↔ ⊃⊃p⊃qr⊃q⊃pr

(alors que : ⊃⊃p⊃qr⊃⊃pq⊃pr + ⊃⊃⊃pq⊃pr⊃⊃s⊃tsu⇒⊃⊃qru ↔ Echec)

Il peut y avoir aussi commutativité de l'application d'un Mthéorème et d'un MMthéorème :

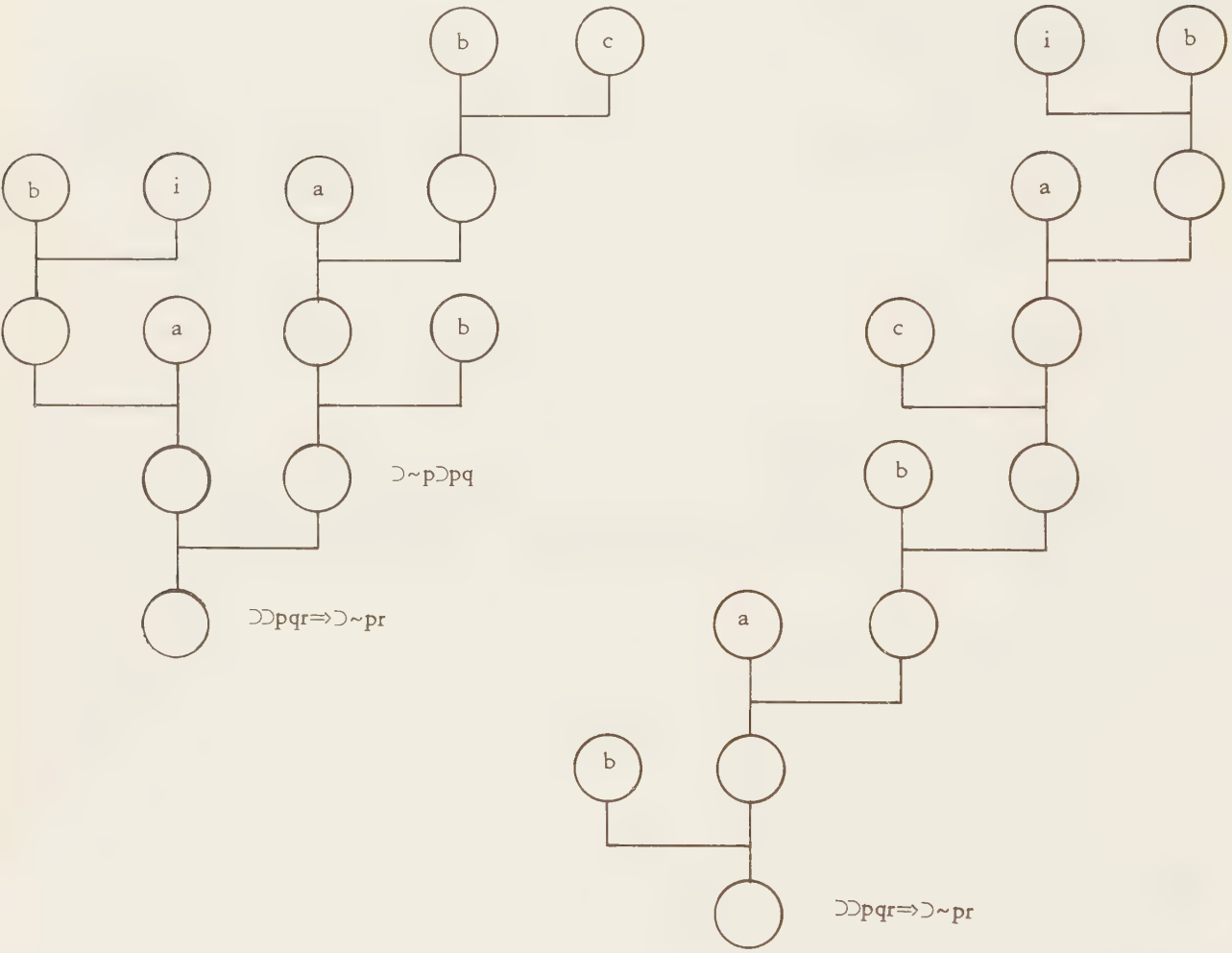
Si A⟶B⇒C transforme les formules selon : α⟶f(α) et si D⇒E a pour formule βiγ, on peut partir du théorème A de formule α, lui appliquer le MMthéorème et l'on a le Mthéorème B⇒C de formule f(α). Si on lui applique enfin D⇒E on aura un Mthéorème de formule βf(α)γ.

Inversement on peut appliquer au théorème A la production D⇒E et l'on aura un théorème de βαγ. Si on applique à ce dernier le MMthéorème, on aura une production de formule f(βαγ).

Les deux formules de démonstration : $\beta f(\alpha)\gamma$ et $f(\beta\alpha\gamma)$ sont très différentes. Il arrive parfois qu'elles correspondent au même énoncé.

Exemple : On a le MMthéorème : $\supset pq \longrightarrow \supset qr \Rightarrow \supset pr$ et la production $\supset\supset pqr \Rightarrow \supset qr$. Dans les deux cas, à tout théorème de la forme $\supset\supset pqr$ il correspond le Mthéorème $\supset rs \Rightarrow \supset qs$. Avec $\supset\supset\sim p\sim q\supset qp$ on a par exemple le Mthéorème : $\supset\supset\sim pqr \Rightarrow \supset qr$.

Nous allons donner les deux arbres correspondants dans l'axiomatique 1. On remarquera comme leur structure est différente. i indique où doit se placer l'antécédent.



4B4 - Conclusion

On a vu les deux raisons pour lesquelles le programme pouvait obtenir plusieurs fois le même énoncé.

Il est possible que, par suite de l'utilisation des Mthéorèmes, on fasse deux dérivations

correspondant à la même formule de démonstration. Il ne s'agit pas d'une véritable ambiguité et le programme a un dispositif qui élimine la plupart des essais correspondants.

Il arrive aussi qu'à un énoncé correspondent plusieurs démonstrations. Cette ambiguité venant de l'axiomatique, on ne peut la supprimer. Les trois grandes causes sont :

- existence de productions neutres du type $A \Rightarrow A$ que le programme va éliminer.

- existence fréquente de dérivations du type :

$$A+B \Rightarrow C \dashrightarrow A$$

- existence fréquente d'une commutativité dans l'ordre d'application des Mthéorèmes ou des MMthéorèmes.

Tout ceci n'a pas des conséquences bien graves. Cela oblige seulement le programme à vérifier avec soin quand il a un résultat nouveau s'il n'existe pas déjà tel que ou sous une forme plus forte (et il est alors sans intérêt) ou sous une forme plus fabile (et il faut supprimer l'ancien résultat).

4C - L'APPRENTISSAGE

4C1 - Généralités

Le mot "apprentissage" est beaucoup trop général. Il est lié à un comportement et non à un mécanisme. On dit qu'il y a apprentissage quand on constate une amélioration des performances, quelqu'en soit la cause. Cette amélioration peut se manifester de diverses façons : dans le cas d'un programme de démonstration de théorèmes :

- le programme peut démontrer des conjectures qu'il ne pouvait démontrer auparavant.
- le programme peut les démontrer plus rapidement.
- la démonstration trouvée est plus simple ou plus élégante.

Malheureusement ces améliorations sont difficilement mesurables.

Ce point de vue behavioriste nous intéresse assez peu. Le problème est de trouver quels mécanismes peuvent conduire à une amélioration des performances. A priori il y a deux possibilités d'action :

- modification de l'information dont le programme peut disposer.
- modification du programme.

a - modification de l'information dont le programme peut disposer

Utilisé sous une forme très grossière, cela conduit à l'apprentissage par coeur : on garde les uns à la suite des autres tous les résultats rencontrés. Cette forme d'apprentissage a peu d'intérêt, il lui manque les deux fonctions de sélection et de structuration dont nous verrons plus loin l'intérêt.

On peut modifier l'information soit en mémorisant de nouveaux résultats, soit en reprenant ce l'on a en mémoire, mais sans rien ajouter de nouveau (c'est une sorte de rumination).

- Mémorisation de nouveaux résultats. Il est bien évident que si le programme a davantage de connaissances, il aura de meilleures performances. S'il faut démontrer des théorèmes, plus il aura de théorèmes à sa disposition et plus il aura de chance d'avoir déjà démontré une conjecture qu'il se

pose. Il est assez difficile de vouloir démontrer ⊃p⊃⊃pqq au départ de l'axiomatique 1. C'est très facile si l'on a déjà ⊃pp et ⊃p⊃qr⇒⊃q⊃pr. Nous n'insisterons pas plus sur l'avantage d'avoir une grande quantité d'information à sa disposition, ce qui est évident. Nous parlerons plutôt des modalités qui doivent accompagner cette augmentation : sélection et structuration.

- Sélection. Il ne faut pas garder tous les résultats que l'on rencontre. Certains doivent être éliminés car ils encombreraient inutilement la mémoire limitée dont on dispose et conduiraient à des essais qui auraient peu de chance de se révéler fructueux : on aurait perte de temps et un encombrement encore plus grand de la mémoire avec tous les résultats inutiles qu'ils produiraient.

- Structuration. Si l'information est emmagasinée n'importe comment, suivant l'ordre dans laquelle on l'obtient, on risque de ne pas retrouver ce dont on a besoin ou d'y mettre un temps prohibitif. A la limite, on se trouve dans le cas de l'enfant qui a appris sa leçon par coeur et qui est incapable de répondre à une question portant sur ce qu'il a appris. L'information doit être disposée de telle façon qu'on ait facilement accès à ce dont on a besoin. Il faut une très forte structuration avec de nombreux liens entre des éléments analogues. La mémoire associative de l'homme est un bon exemple de ce qu'il faudrait réaliser. La façon dont on mémorise, la facilité avec laquelle on peut accéder à l'information désirée, la richesse des associations que la mémoire va favoriser comptent autant que la qualité de l'information que l'on garde.

Le programme doit choisir ce qu'il garde, le ranger de telle façon qu'il puisse le retrouver facilement quand il en aura besoin, et pour cela le lier aux informations de nature voisine qui sont déjà en mémoire.

- Rumination. Ce travail peut se faire sans que l'on ait d'éléments nouveaux. On améliore ce que l'on a sans augmenter la quantité d'information dont on dispose. Trois procédés au moins sont possibles :

- Sélection. On élimine des résultats qui ont cessé d'avoir de l'intérêt ou qui se sont révélés moins intéressants qu'on ne le pensait a priori. On est souvent géné par un excès de résultats. Il est souvent intéressant de faire une sélection assez grossière au moment où on trouve un résultat et d'éliminer par la suite ceux qui ont eu peu d'utilité.

- Structuration. On reclasse des résultats mal placés initialement, on rajoute des connexions entre résultats, on enlève des connexions inutiles.

- Précision. L'information archivée comporte des symboles, mais aussi des paramètres dont la valeur va influer sur le rôle que l'on fera jouer aux symboles qui leur sont liés. Ce peut être l'intérêt d'un résultat, ceux qui ont un intérêt élevé ayant plus de chance de participer à un essai. Une estimation plus précise de la valeur de ces paramètres amènera une amélioration des performances.

L'augmentation de l'information dont le programme peut disposer a une importance évidente. Mais il faut savoir choisir ce que l'on garde et le lier à ce que l'on connait déjà pour en faciliter l'utilisation. Il faut par la suite y revenir, s'assurer qu'il mérite toujours d'être gardé et modifier s'il y a lieu les liaisons avec les autres éléments.

b - modification du programme

Elle peut se faire à différents niveaux :

- modification de la valeur des paramètres qui figurent dans le programme. Il y a là simplement une adaptation (qui peut avoir de grandes conséquences si l'on modifie la valeur du seuil qui décide si l'on archive un résultat).

Cela fonctionne comme un servomécanisme. A chaque paramètre est associé une grandeur de commande dont la valeur déterminera la modification. Si par exemple K est le paramètre et C la grandeur de commande, on peut prendre le schéma suivant :

$C<L_1$	K devient $K-K_1$
$L_1<C<L_2$	K devient $K-K_2$
$L_2<C<L_3$	K ne change pas
$L_3<C<L_4$	K devient $K+K_2$
$L_4<C$	K devient $K+K_1$

Si K_1 et K_2 sont positifs, le système sera stable si une augmentation de K tend à diminuer C. Bien entendu $K_1>K_2$. On a mis 5 niveaux. On aurait pu en mettre plus, ou même une correction continue du type K devient $K+a(C-C_d)$ qui risque de donner une correction trop forte si $C-C_d$ devient très grand : on peut avoir des oscillations sur la valeur de K. On pourrait avoir le même phénomène si l'on avait pris 3 niveaux au lieu de 5, par suite dans ce cas du manque de finesse de la réaction.

Cette méthode demande qu'il y ait une corrélation assez forte entre la valeur de K et celle que va prendre C. Sinon K va prendre n'importe quelle valeur. A titre d'exemple, on peut prendre pour K le seuil qui décide si on archive un résultat et C le nombre de résultats archivés. Dans ce cas on tendra à archiver un nombre de résultats compris entre L_2 et L_3 (ou voisin de C_d dans le cas de la correction continue). Si on archive trop de résultats, on augmente le seuil et on en archivera moins.

Cette possibilité de faire évoluer les valeurs des paramètres est importante. Dans certains cas, on ne sait pas quelle valeur leur donner. Il suffit de choisir une valeur approximative que le programme va ajuster. Dans d'autres cas la valeur du paramètre doit évoluer au cours du temps parce que la situation change. Cela se produit souvent dans l'étude des axiomatiques où le démarrage est pénible et qui deviennent faciles à partir du moment où l'on a trouvé un résultat clé. Un très bon exemple est donné par l'axiomatique 2, le résultat clé étant T 6.

- modification de la structure du programme.

Il s'agit essentiellement de lui faire utiliser de nouvelles heuristiques. Ce qui suit est largement hypothétique et n'a pas été essayé. Cela ne peut servir que de point de départ à une réalisation concrète de ces méthodes.

Le procédé le plus simple qui vienne à l'esprit est de donner au programme la possibilité de changer la nature des paramètres qui commandent son comportement. Par exemple de rajouter dans le calcul de l'intérêt syntaxique un terme relatif au nombte de connectives binaires suivies par une connective unaire. Une puissance plus grande serait obtenue si le programme pouvait changer de méthode : par exemple au lieu de prendre le résultat qui a l'intérêt le plus élevé et l'essayer avec tous les résultats d'une arborescence, on pourrait n'essayer que le couple de résultats auquel correspond l'essai ayant l'intérêt le plus élevé.

Pour réaliser ceci il faut un "super programme" qui domine le programme de performance, analyse ses résultats et dispose d'heuristiques de niveau plus éleve (qui ont avec les heuristiques deja rencontrées une relation analogue à celle des Mthéorèmes avec les théorèmes) pour modifier le programme. On peut garder la liste des paramètres qui ont été essayés dans le passé. A chaque essai, réussi ou non, le superprogramme regarde quels sont les paramètres qui ont contribué le plus à le choisir. On garde ceux qui ont un bilan positif et on élimine ceux qui se révèlent trop souvent nocifs pour les remplacer par d'autres paramètres de la liste.

Changer de méthode semble impossible à réaliser dans l'état actuel de nos connaissances. Il faudrait pouvoir disposer d'un langage assez puissant pour exprimer les heuristiques et pour les modifier. Il y a là une voie de recherche que l'on n'a pas encore abordé.

4C2 - Possibilités d'apprentissage du programme

a - modification de l'information dont le programme peut disposer

Au moment de la mémorisation d'un nouveau résultat il y a sélection qui, on l'a vu, se fait de deux façons :

- l'intérêt du résultat doit être supérieur à un seuil. La façon de calculer l'intérêt est à peu près indépendante de l'axiomatique étudiée.

- le résultat peut être sans intérêt parce qu'on a déjà trouvé des résultats qui rendent son emploi inutile. Ceci dépend essentiellement de ce que l'on a déjà trouvé et tel résultat éliminé dans une axiomatique peut être très utile dans une autre. On a ainsi la possiblité de s'adapter efficacement au problème particulier que l'on traite.

Le résultat que l'on garde est archivé dans une arborescence ce qui permet de le retrouver très rapidement si l'on en a besoin.

Une fois le résultat archivé, le programme peut l'éliminer par la suite s'il estime qu'il n'en a plus besoin. Ce peut être parce que son intérêt est devenu trop faible par suite de l'influence de l'intérêt historique, ou parce que le seuil d'archivage a augmenté. Ce peut être parce que l'on a trouvé un nouveau résultat qui rend inutile qu'on le garde.

Cette élimination de résultats que l'on avait archivé peut avoir des conséquences très importantes et dans certains cas le nombre des résultats archivés, puis effacés peut dépasser celui des résultats qui restent en mémoire à la fin de l'essai. A titre d'exemple nous allons donner des chiffres pour quelques exploitations. La première colonne donne le nombre de résultats archivés à la fin de l'essai, la deuxième de ceux archivés, puis effacés. Ce dernier nombre est la somme de deux nombres : résultats éliminés parce que leur intérêt a été jugé par la suite trop faible qui est dans la troisième colonne et résultats éliminés par la découverte d'autres résultats qui est dans la quatrième.

	résultats qui restent en fin d'essai	résultats archivés puis effacés	parce que leur intérêt a été jugé trop faible	éliminés par la découverte d'autres résultats
Axiomatique 1	180	45	35	10
Axiomatique 3	170	126	6	120
Axiomatique 5	228	309	285	24
Axiomatique 6	228	62	61	1

Ceci a un rôle très important, car conserver trop de résultats encombre la mémoire et conduit à faire des essais inutiles.

Il n'y a guère de modification de la structuration de la mémoire. On se contente de mémoriser, liée à l'énoncé d'un résultat, la localisation des résultats avec lequel on l'essaye et celle des couples de résultats qui le produisent.

Il y a enfin amélioration de la précision des paramètres liés à l'information grâce à l'intérêt historique. Cela améliore le comportement du programme : meilleur choix des résultats à essayer et meilleure sélection des essais.

b - modification du programme

Elle est très sommaire. On se contente d'adapter quelques paramètres. Mais l'un de ces paramètres, le seuil d'archivage a une très grande importance sur le comportement du programme. La grandeur de commande est le rapport N^2/E où N est le nombre de résultats archivés et E le nombre de résultats que l'on a démontré. On a pris N^2 au lieu de N, car il est normal d'archiver davantage de résultats en début d'étude et de se montrer plus sévère par la suite. Si la grandeur de commande devient trop grande, on augmente le seuil et inversement. On a laissé s'adapter deux autres paramètres, mais ils ont beaucoup moins d'importance. On n'a pas donné au programme la possibilité d'opérer sur lui-même des modifications plus profondes.

4C3 - Conclusion

L'apprentissage a un rôle capital dans le fonctionnement du programme. Il est essentiellement dû à l'augmentation sélective et structurée de l'information dont il peut disposer. Je ne suis pas arrivé à maitriser la deuxième méthode où le programme se modifie lui même et d'un programme à l'autre je lui ai donné de moins en moins d'importance. Par contre, la sélection et l'archivage des résultats s'est fait en utilisant des méthodes de plus en plus élaborées.

La deuxième méthode aurait pourtant un avantage capital : l'apprentissage qu'elle donnerait pourrait se conserver en partie de l'étude d'une axiomatique à celle d'une autre axiomatique. On pourrait augmenter son habilité à traiter des axiomatiques. Dans ce qui a été réalisé, essentiellement à base d'archivage, les performances croissent de façon très nette au fur et à mesure que le programme travaille une axiomatique particulière. Mais ce qu'il apprend ainsi ne peut lui servir quand il commence une nouvelle axiomatique.

Si le programme avait la possibilité de créer des heuristiques, l'expérience acquise sur une axiomatique serait partiellement valable avec une axiomatique nouvelle. Malheureusement personne n'a, jusqu'à présent, résolu ce problème de façon efficace.

4D - DEMONSTRATION DE CONJECTURES

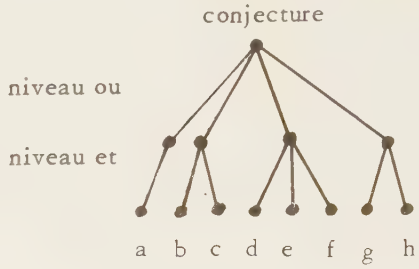
4D1 - L'arbre "ou - et"

On a voulu donner au programme la possibilité de travailler de manière classique : on lui donne un énoncé et il doit trouver une suite de dérivations qui relient les axiomes à cette conjecture.

On utilise la méthode suivante : supposons que l'on veuille démontrer la conjecture A. Si $B \Rightarrow A$ est une production ou se déduit d'une production par des substitutions, si l'on a démontré B, on aura démontré A. On s'est donc ramené à un nouveau problème : démontrer B. Ceci n'est intéressant que si B est plus facile à démontrer que A. Cette méthode est proche du "detachment" exposé dans les articles de Newell, Shaw et Simon cités en 1A. Ils utilisent l'implication de la langue au lieu de celle de la Mlangue. Ce n'était pas gênant pour eux qui voulaient étudier un seul système formel, mais c'est moins général : il se peut que la langue n'ait pas d'implication. D'autre part on peut avoir $B \Rightarrow A$ sans avoir $\supset BA$ alors que, si l'on a le modus ponens, $\supset BA$ entraine $B \Rightarrow A$. Ce procédé va produire ce que l'on appelle un arbre "ou - et". Supposons que nous ayons la conjecture A et les productions $a \Rightarrow b$ et $c,d \Rightarrow e$. Si par des substitutions ces productions deviennent $B \Rightarrow A$ et $C,D \Rightarrow A$, on voit que A sera démontré si l'on a démontré B **ou** si l'on a démontré C **et** D. On voit apparaitre la structure ou - et. Bien entendu il peut y avoir un nombre quelconque de "ou" et de "et". Il y a autant de résultats liés par "ou" que de productions essayées et pour chaque production à n antécédents n conjectures liées par "et". A leur tour on peut appliquer à ces nouvelles conjectures

le même procédé et l'on aura à une arborescence où un niveau sur deux correspondra à un "ou", l'autre correspondant à un "et".

Pour démontrer une conjecture il faut et il suffit que l'on ait démontré toutes les conjectures qui dépendent d'un de ses noeuds "ou".
Exemple. La conjecture sera démontrée si l'on a démontrée a ou (b et c) ou (d et e et f) ou (g et h). Si l'on a démontré a, la conjecture est démontrée, mais elle ne l'est pas si l'on n'a démontré que b,d,e et g.



Remarque 1 - On applique les productions à l'envers. On a déjà vu en 2B (Remarque 6) qu'il faut une légère modification à l'algorithme de dérivation : on fait coïncider le résultat et la conjecture, mais sans avoir le droit de faire des substitutions dans la conjecture. Sinon on démontrerait un résultat plus faible. On avait créé le signe $\twoheadrightarrow$ pour représenter cette opération.

Remarque 2 - En plus des productions il est souvent utile d'appliquer une définition qui mette la conjecture sous une forme moins compacte : on pourra lui appliquer alors d'autres productions. Si l'on cherche à démontrer dans l'axiomatique 1 $C\wedge pqp$, en lui appliquant la définition, on a : $C\sim Cp\sim qp$ et on peut alors essayer la production : $C\sim pq\Rightarrow C\sim qp$.

Remarque 3 - Le programme ne peut prendre comme conjecture que des théorèmes. On aurait pu facilement lui donner la possibilité de démontrer des Mthéorèmes : on appliquerait la même méthode en faisant agir des MMthéorèmes :

$Cp Cqr\Rightarrow Cq Cpr + Cpq \longrightarrow p\Rightarrow q \twoheadrightarrow CCp Cqr Cq Cpr$

$Cpq, Cqr\Rightarrow Cpr + p\Rightarrow Cqr \longrightarrow p, q\Rightarrow r \twoheadrightarrow Cpq\Rightarrow CCqr Cpr$

La première ligne montre que si l'on veut démontrer $Cp Cqr\Rightarrow Cq Cpr$, il suffit de démontrer $CCp Cqr Cq Cpr$.

Par contre, il faudrait une modification très profonde pour lui permettre d'essayer de démontrer des MMthéorèmes. Il faudrait en particulier lui donner des moyens dans la MMMlangue qu'il n'a pas. Nous avons déjà abordé ce point en 2E.

Exemple : Donnons un exemple d'arbre "ou - et". Supposons que l'on veuille démontrer : $E\wedge\wedge pqr\wedge p\wedge qr$ dans l'axiomatique 1.

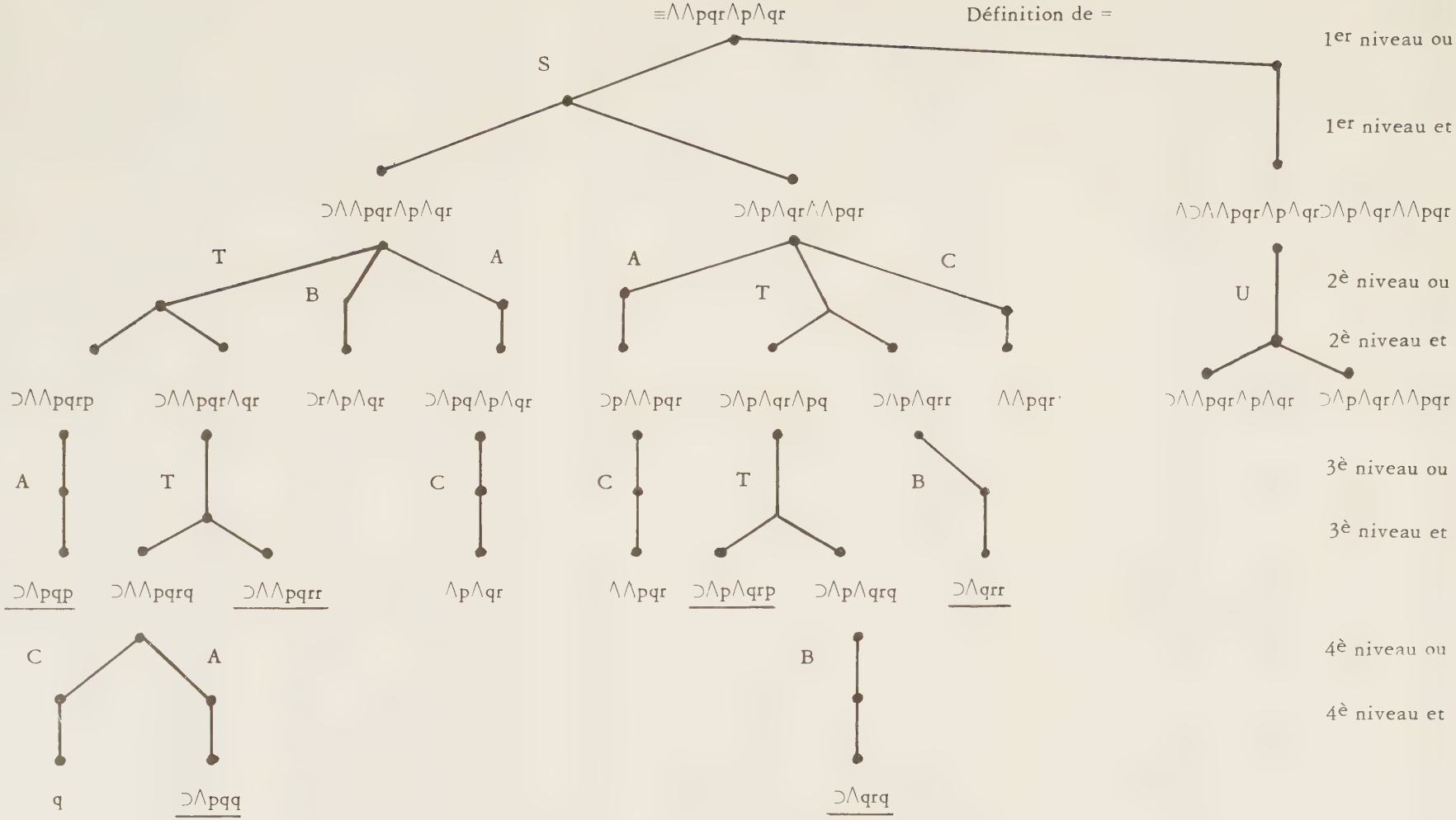
On a déjà les Mthéorèmes :

S : $Cpq, Cqp\Rightarrow Epq$
T : $Cpq, Cpr\Rightarrow Cp\wedge qr$
U : $p, q\Rightarrow\wedge pq$
A : $Cpq\Rightarrow C\wedge prq$
B : $Cpq\Rightarrow C\wedge rqp$
C : $p\Rightarrow Cqp$

et les théorèmes : $C\wedge qp$ et $C\wedge pqq$

Nous soulignons les sous conjectures qui sont des théorèmes déjà connus. Les lettres indiquent la production utilisée.

On voit quelle est la démonstration : $C\wedge pqp$ étant un théorème, $C\wedge\wedge pqrp$ en est un ainsi que


≡ΛΛpqrΛpΛqr
Définition de =
1er niveau ou
S
1er niveau et
ɔΛΛpqrΛpΛqr
ɔΛpΛqrΛΛpqr
ΛɔΛΛpqrΛpΛqrɔΛpΛqrΛΛpqr
T
B
A
A
T
C
U
2è niveau ou
2è niveau et
ɔΛΛpqrp
ɔΛΛpqrΛqr
ɔrΛpΛqr
ɔΛpqΛpΛqr
ɔpΛΛpqr
ɔΛpΛqrΛpq
ɔΛpΛqrr
ΛΛpqr
ɔΛΛpqrΛpΛqr
ɔΛpΛqrΛΛpqr
A
T
C
C
T
B
3è niveau ou
3è niveau et
ɔΛpqp
ɔΛΛpqrq
ɔΛΛpqrr
ΛpΛqr
ΛΛpqr
ɔΛpΛqrp
ɔΛpΛqrq
ɔΛqrr
C
A
B
4è niveau ou
4è niveau et
q
ɔΛpqq
ɔΛqrq

⊃Λp Λqrq et ⊃Λp Λqrp (ce dernier en opérant une substitution). De même de ⊃Λpqq, on déduit ⊃ΛΛpqrq, ⊃ΛΛpqrr et ⊃Λp Λqrr.

Mais, avec ⊃ΛΛpqrq et ⊃ΛΛpqrr, on a grâce à T : ⊃ΛΛpqrΛqr.

De même, avec ⊃ΛΛpqrp et ⊃ΛΛpqrΛqr, on a : ⊃ΛΛpqrΛp Λqr

On verrait de la même façon que l'on a : ⊃Λp Λqr ΛΛpqr.

Et S avec ces deux derniers résultats donne la conjecture désirée.

Remarque 4 - Pour désigner un point de l'arborescence nous donnerons la suite des numéros des branches qu'il faut prendre pour y arriver à partir de la conjecture sous la forme ---(i,j)---. i étant le numéro d'une branche "ou" et j celui d'une branche "et".

⊃pΛΛpqr se trouve en : (1,2) (1,1)
⊃Λqrr en (1,2) (2,2) (1,1)

Remarque 5 - Nous voyons sur l'exemple que l'on a donné qu'un même énoncé peut figurer en divers endroits : ⊃ΛΛpqrΛp Λqr se trouve à la fois en (1,1) et en (2,1) (1,1). Le programme regarde à chaque nouvelle conjecture s'il ne l'a pas déjà rencontrée pour éviter de faire deux fois les mêmes essais. De même quand il a réussi à démontrer une sous conjecture, il regarde si elle ne se trouve pas ailleurs.

4D2 - Mise à jour de l'arbre

a - En cas d'échec

Une sous conjecture est considérée comme ne pouvant être démontrée, donc comme conduisant à un échec si :

- Elle se réduit à une variable. Dans l'exemple que l'on avait donné, q en (1,1) (1,2) (1,1) (1,1). Une variable isolée ne peut être un théorème, ou alors tous les énoncés formés en suivant les règles de la syntaxe seraient des théorèmes.

- Elle dépasse une certaine taille. Soient u et b les nombres de connectives unaires et binaires. Si Au+Bb>C, on considérera qu'il est inutile d'essayer de démontrer la sous conjecture. A,B,C sont des constantes données. Ceci n'a qu'une valeur heuristique et traduit le fait qu'en général on n'a pas besoin de passer par des intermédiaires très longs pour démontrer une conjecture. Il est pourtant possible que, dans certains cas, cela élimine une étape indispensable.

- On ne peut lui appliquer ni productions, ni définitions. Si on ne peut agir sur la sous conjecture, il n'y a pas d'espoir qu'elle donne naissance à d'autres sous conjectures qui finalement se ramèneront à des théorèmes connus.

- Si toutes les sous conjectures auxquelles elle a donné naissance ont conduit à des échecs. On voit qu'un échec va se propager vers le haut de l'arbre "ou - et". La branche "ou" dont dépend l'échec doit être supprimé. En effet, même si l'on démontre les autres branches "et", il en manquera une. S'il ne reste alors plus aucune branche "ou" à la sous conjecture du niveau supérieur, on applique le même procédé. Si on arrive finalement à la conjecture de départ, cela signifie que l'on n'a pas pu la démontrer avec les moyens dont on disposait. Il se peut, même en cas d'échec, qu'elle soit quand même un théorème. La méthode utilisée est heuristique ; sa faillite ne signifie pas qu'il n'existe pas de dérivation, mais qu'on n'a pas su la trouver.

Remarque - On considérera qu'une branche (et non un résultat) de l'arborescence est sans intérêt si

la sous conjecture qui est à son extrémité inférieure figure déjà parmi ses ascendants. Il est bien évident que la réussite à ce niveau n'apportera rien : elle peut tout au plus amener à montrer qu'elle est un théorème, ce que l'on sait déjà.

b - En cas de réussite

On vient de voir qu'une sous conjecture figure dans les théorèmes ou peut se déduire d'un théorème par des substitutions. S'il y a d'autres branches au niveau "et" immédiatement supérieur, on se contente d'enlever la branche dont elle dépend et il reste à démontrer les résultats liés aux autres branches. S'il n'y a pas d'autre branche, la sous conjecture immédiatement au-dessus est à son tour démontrée (même s'il reste d'autres branches "ou") et il suffit de lui appliquer la même procédure. Si c'est la conjecture de départ, le programme a résolu le problème qui lui était posé.

Les deux modes de mise à jour sont duals l'un de l'autre.

4D3 - Utilisation du procédé

Le sous programme qui traite les procédures fonctionne correctement. Il a été assez peu testé pour diverses raisons :

- il occupe une place en mémoire importante, ce qui diminue d'autant la place réservée aux listes.
- il faut qu'il reste pas mal de place dans la mémoire de listes pour pouvoir créer l'arbre "ou - et" Or on a intérêt à l'utiliser quand le programme a travaillé assez longtemps pour avoir démontré un nombre important de théorèmes et de Mthéorèmes. Ceux qu'il pourrait démontrer dès le début sont assez rapidement découverts par le programme sans qu'il soit nécessaire de lui en indiquer l'existence. Malheureusement, quand le programme a démontré beaucoup de résultats ce qui permettrait de trouver des démonstrations difficiles, il ne reste plus assez de place pour construire un arbre "ou - et" important.

Finalement la méthode n'a été sérieusement essayée qu'avec l'axiomatique 6. Il est probable qu'une étude plus complète aurait amené à modifier et à compléter les heuristiques que l'on a données.

Il manque au programme un dispositif qui rende plus intéressante l'utilisation de cette méthode : c'est la possibilité de définir lui même les conjectures. Il est certain que le mathématicien se pose des sous problèmes qu'il essaye de démontrer. Il est difficile de le programmer, car il semble qu'on utilise des notions sémantiques : dans une interprétation, on observe une "loi expérimentale" et on essaye de la prouver formellement. Le programme de GELERNTER fait une utilisation intéressante d'un modèle, mais le système formel qu'il étudie a une interprétation géométrique. Si sur une figure il semble que deux angles sont égaux, il va essayer de le démontrer. Comme nous n'avons pas fait de sémantique, nous ne pouvons prendre ces méthodes. Une solution serait de conjecturer des Mthéorèmes, c'est à dire des lois de transformation des suites de symboles qui sont des théorèmes. Par exemple tel symbole est-il commutatif : a-t-on $\wedge pq \Rightarrow \wedge qp$?

Il y a là certainement une lacune importante. Il est malheureusement difficile de voir comment on pourrait la combler. De toutes les façons, notre but est de faire de l'intelligence artificielle, et non de la simulation de l'intelligence, nous nous sommes libérés de la contrainte : opérer comme le ferait un être humain.

CHAPITRE V

PROGRAMMATION

5A - LES TROIS PREMIERS PROGRAMMES

J'ai écrit, mis au point et exploité quatre programmes. Le dernier sera spécialement étudié en 5C. Nous allons donner ici rapidement les caractéristiques les plus importantes des trois premiers.

5A1 - Le premier programme

Ecrit, mis au point et exploité en 1961 sur le 7090 du Centre de Calcul Scientifique d'I.B.M. Il comprenait 800 instructions Fortran.

Une de ses particularités était de faire largement appel au hasard pour choisir les essais qu'il devait tenter. Cela s'est révélé catastrophique : d'une part on répétait des essais, d'autre part on tardait à faire des essais importants. Par la suite on n'a plus jamais utilisé des nombres aléatoires et les trois derniers programmes sont entièrement déterminés.

On avait également un dispositif d'apprentissage qui avait été inspiré par l'essai remarquable de SAMUEL pour le jeu de dames (Some studies in machine learning using the game of checkers. I.B.M. Journal of research and development Vol. 3 N° 3). On mesurait pour chaque théorème les valeurs c_i d'un certain nombre de caractéristiques (par exemple nombre de connectives unaires ou nombre de variables différentes). On définissait l'intérêt par $I = \Sigma c_i d_i$, les d_i étant des paramètres que l'apprentissage devait modifier. Le seul schéma de dérivation utilisé étant : $a;b;a,b\Rightarrow c \longrightarrow c$, on faisait intervenir dans I les caractéristiques de a et de b. Si I était supérieur à un certain seuil, on faisait l'essai et on avait un résultat c dont on évaluait l'intérêt J. On modifiait les d_i en fonction de "l'erreur" : I-J. Si cette méthode avait fonctionné correctement, cela aurait permis d'évaluer avec une assez bonne probabilité l'intérêt d'un essai avant de l'effectuer. Malheureusement cela a totalement échoué, probablement en partie à cause d'un mauvais choix des d_i et de leur formule de correction en fonction de I-J. Ce type d'apprentissage a été abandonné par la suite.

Le programme n'a été essayé qu'avec l'axiomatique 1. Le seul résultat qu'il ait réussi à démontrer a été $\supset pp$! Il s'est donc révélé très peu performant. Il était trop lié à l'axiomatique 1 et n'utilisait pas de Mthéorèmes. Son principal intérêt a été de montrer les dangers de l'utilisation inconsidérés de l'aléatoire et de la difficulté d'avoir un apprentissage efficace.

5A2 - Le deuxième programme

Mis au point en 1962 sur le 7090 du Centre de Calcul Scientifique d'I.B.M., il a été exploité la même année sur l'I.B.M 704 de l'Institut Blaise Pascal. Il comprenait 1000 instructions Fortran.

Il ne démontrait pas de Mthéorèmes, mais il distinguait les théorèmes intéressants à faire coïncider avec le premier antécédent de $p,\supset pq\Rightarrow q$ de ceux utiles avec le deuxième antécédent. Pour ces derniers, il évaluait l'intérêt en comparant la partie correspondant à p avec celle correspondant à q. Cela revenait à utiliser le MMthéorème $\supset pq \longrightarrow p\Rightarrow q$ et à déduire l'intérêt du théorème de celui du Mthéorème correspondant. De plus on se limitait à un type de production bien déterminé, alors que l'utilisation de $a,b\Rightarrow c \rightsquigarrow b \longrightarrow a\Rightarrow c$ offre des possibilités beaucoup plus vastes.

L'intérêt d'un résultat était évalué en cherchant à quel type il appartenait ; le type était obtenu à partir de l'énoncé du théorème, B désignant une connective binaire, U une unaire et V une variable ou un terme.

Exemple : ~~p est du type : UUV ou UV ou V

CCpCqrCCpqCpr est du type : BBVBVVBBVVBVV ou BBVVBVV ou BVBVV ou BBVVV ou BVV ---

Parmi les types possibles, le programme en choisissait un ou plusieurs d'après des critères heuristiques tenant compte de la taille des termes que l'on devait substituer aux diverses occurrences de V pour retrouver le théorème. Ce mécanisme était assez efficace pour sélectionner les essais, car il n'essayait un théorème d'un certain type qu'avec un théorème du type Cpq dont la partie correspondant à p était du même type.

Le programme avait la possiiblité de voir si un résultat nouveau n'était pas plus faible qu'un résultat déjà démontré. Avec l'axiomatique 1 il a démontré :

CCpqCCrpCrq
Cpp
C~pCpq
C~~pp
Cp~~p
CpCCpqq
CCpqCCqrCpr
CCCpqrCqr
CpC~pq

Ce qui est satisfaisant, mais, essayé avec les axiomatiques 2 et 6 il n'a à peu près rien trouvé les heuristiques étaient trop bien adaptées à l'axiomatique 1. Ce sont les démonstrations qu'il a trouvées qui m'ont amené à découvrir l'importance des Mthéorèmes.

Exemple 1

Démonstration trouvée par le programme

CC~p~qCqp
CrCC~p~qCqp
CCrC~p~qCrCqp
CpCqp
C~pCpq

CpCqp
CCpCqrCCpqCpr

Démonstration donnée par Rosenbloom

CC~p~qCqp
CCrC~p~qCrCqp
CpCqp
C~pCpq

CCpqCCrpCrq

On voit que cela revient à utiliser Cpq⇒CCrpCrq pour les deux premiers pas et CCpqr⇒Cqr pour l'ensemble des trois.

Exemple 2 Je donne uniquement la démonstration du programme

CCpqCCrpCrq
CCCpqCrpCCpqCrq
CCsCCpqCrpCsCCpqCrq
CpCqp
CCrpCCpqCrq

CCpCqrCCpqCpr
CCpqCCrpCrq

soit CCpqCCqrCpr. Cela revient à utiliser CpCqr⇒CqCpr (sous une forme assez peu élégante). Rosenbloom démontrait d'abord CCpCqrCqCpr qui est assez difficile à démontrer.

C'est en voyant que le programme utilisait souvent le schéma :

Résultat 1 $Cp Cqp$

Résultat 2

$CCpCqr\ CpqCpr$

Résultat 3

alors que Rosembloom utilisait :

Résultat 1 $CCpqCCrpCrq$

Résultat 3

que j'ai compris l'importance des Mthéorèmes et que j'ai décidé d'essayer par la suite de lui donner la possibilité de les manipuler.

En résumé le deuxième programme qui avait déjà des résultats intéressants avec l'axiomatique 1 a eu pour principal intérêt de m'aider à dégager l'importance de la possibilité de créer des Mthéomes.

5A3 - Le troisième programme

Mis au point et exploité sur l'I.B.M. 704 de l'Institut Blaise Pascal. Il comprenait 1800 instructions Fortran.

Ce programme pouvait démontrer des Mthéorèmes et des MMthéorèmes. Mais il n'utilisait pas, pour calculer l'intérêt des théorèmes, l'intérêt des Mthéorèmes auxquels ils pouvaient donner naissance. Il utilisait une méthode analogue à celle du deuxième programme qui servait aussi à sélectionner les essais. Malheureusement le nombre de types que le programme connaissait étant limité, il avait des difficultés dans les axiomatiques où les résultats comportent beaucoup de symboles ce qui augmente la variété de types possibles (par exemple les axiomatiques 3 et 5). L'utilisation des Mthéorèmes n'était pas satisfaisante, en grande partie par suite de l'absence de l'étude exposée en 3B3. Il y avait beaucoup trop de productions sans intérêt. Les résultats n'étant pas archivés en arborescence, une telle recherche aurait pris un temps prohibitif : on se contentait de regarder pour chaque nouveau résultat s'il n'était pas déjà archivé sous une forme plus forte ou plus faible.

Avec l'axiomatique 1, le programme a redémontré la plupart des résultats trouvés par le deuxième programme et quelques résultats nouveaux comme :

$CCpqCpCrq$
$CCpCqrCqCpr$
$C\sim Cpqp$
$C\wedge pqp$
$C\equiv pqCpq$
$C\wedge pqq$

Donnons la démonstration de $CCpCqrCqCpr$ qui est différente à la fois de celle de Rosenbloom et de celle du quatrième programme. On veut enlever le deuxième p de $CCpCqrCCpqCpr$. Pour cela on fait passer en tête un terme qui le contient, on enlève p et on revient à l'ordre initial :

$CCpCqrCCpqCpr + CpCqr \Rightarrow CqCpr \dashrightarrow CCpqCpCqrCpr$
$CCpqCpCqrCpr + CCpqr \Rightarrow Cqr \dashrightarrow CqCpCqrCpr$
$CqCpCqrCpr + CpCqr \Rightarrow CqCpr \dashrightarrow CCpCqrCqCpr$

Il y avait malheureusement beaucoup de résultats sans intérêt qui masquaient les résultats intéressants et asphyxiaient rapidement le programme.

Avec d'autres axiomatiques, les résultats étaient très faibles. Avec l'axiomatique 6, les seuls théorèmes intéressants qu'il a pu démontrer étaient :

$\supset pp$

$\supset p \vee pq$

Le programme avait la possibilité de démontrer des conjectures, mais on n'a pas fait des essais, car les productions archivées étaient trop mal choisies pour que l'on puisse espérer un comportement efficace.

En résumé c'est avec ce programme qu'est apparue la possibilité de démontrer Mthéorèmes et MMthéorèmes. Mais ils n'étaient pas utilisés pour évaluer l'intérêt des théorèmes. D'autre part l'élimination des résultats en tenant compte de ceux déjà trouvés était trop sommaire. Il n'a donné des résultats acceptables qu'avec l'axiomatique 1, mais a permis de voir quelles améliorations il fallait apporter et a conduit ainsi à l'élaboration du quatrième et dernier programme.

5B - UTILISATION DE L'OUTILLAGE DE PROGRAMMATION EN LISTES

- Dans toutes les opérations de dérivation, il est important de pouvoir traiter des chaines, c'est à dire des suites linéaires de symboles concaténés, en effectuant aisément des suppressions, des insertions ou même le remplacement d'un certain groupe par un autre. On a constamment des opérations de ce type au cours des substitutions : Si dans $\supset p \supset qp$ on fait la substitution $q/\sim q$, on aura $\supset p \supset \sim qp$ et il est intéressant de pouvoir insérer $\sim$ entre le deuxième $\supset$ et q sans être obligé de tout réécrire.

- Pour l'archivage des résultats en arborescence (que l'on verra en 5D) et pour la démonstration des conjectures à l'aide des arbres "ou - et", il faut pouvoir représenter des structures très complexes où chaque noeud a un nombre quelconque de branches qui varie au cours du temps et où le nombre de niveaux au dessous d'un noeud peut être aussi grand qu'on le veut.

Un outillage de listes est surpuissant pour le premier besoin, mais indispensable pour le second.

A l'époque de l'écriture du premier programme, au début de 1961, il existait quatre outillages de programmation utilisant les listes : IPL, LISP, COMIT et FLPL.

IPL avait l'inconvénient d'être lent et d'avoir une écriture proche du langage machine ce qui rend son utilisation pénible.

COMIT est également lent. Il est nettement orienté vers la manipulation des lettres pour les études de traduction automatique. Son écriture n'est pas très naturelle pour d'autres applications et il semble mieux s'adapter au traitement des chaines.

LISP est très séduisant, en particulier par sa récursivité. Mais la version interprétative qui existait à l'époque était très lente. La documentation que l'on pouvait avoir était incomplète et il lui manquait des caractéristiques indispensables comme l'arithmétique fixe. Par contre il avait un système de récupération des mémoires devenues libres très intéressant.

FLPL créé par GELERNTER et HANSEN (cf par exemple : A Fortran compiled list processing

language. Journal of the A.C.M. Avril 60 Vol. 7 N° 2 p. 87) fait en quelque sorte la fusion de IPL et de Fortran. Il se présente sous la forme d'un ensemble de fonctions que l'on pourra appeler à partir du programme Fortran exactement comme un sinus ou un logarithme. En schématisant, ces fonctions peuvent se grouper en 5 familles :

- fonctions qui extraient l'information.
- fonctions qui modifient l'information déjà mémorisée.
- fonctions de recherche (par exemple : le dernier élément d'une liste).
- fonctions qui créent de nouvelles listes ou sous listes.
- fonctions de servitude (effacement des listes inutiles, impression des listes ---)

En réalité leur emploi n'est pas tout à fait analogue à celui des fonctions Fortran classiques. Elles sont souvent en même temps des opérateurs : d'une part elles ont une valeur et d'autre part elles modifient le contenu de la mémoire. Souvent on ne s'intéresse même pas à leur valeur. Par exemple la fonction XSTORDF (J,K) a pour effet de mettre la quantité K dans les bits 3 à 17 du mot J. La valeur de la fonction est ce qu'il y avait dans cette zone avant d'y mettre K. Si on veut changer la valeur de cette partie d'un mot sans s'intéresser à ce qu'il y avait avant, on peut écrire : JNK = XSTORFD(J,K) JNK étant une variable dont on n'utilisera pas la valeur par la suite. Mais si l'on écrit :

JNK = XSTORDF(I,XSTORDF(J,XCDRF(I))

la fonction XCDRF(I) ayant pour valeur la quantité contenue dans les bits 3 à 17 du mot I et n'ayant pas d'autre action, on voit que l'on va ainsi échanger le contenu de cette partie du mot I avec la partie correspondante du mot J.

Cet outillage a de nombreuses qualités. C'est pourquoi on l'a utilisé pour les quatre programmes.

- il est disponible à la fois sur l'I.B.M. 704 et l'I.B.M. 7090.

- il est parfaitement au point. Il n'y a pas eu une seule erreur venant de l'outillage.

- il tient peu de place en mémoire. Comme il se compose d'un grand nombre de fonctions, on ne met que celles dont on a besoin ce qui permet de le tailler sur mesure. Dans le cas du quatrième programme l'ensemble des fonctions utilisées occupait 400 mémoires ce qui est très peu, surtout quand on tient compte de la puissance qu'il apporte. Cela était d'autant plus précieux que l'on était à court de mémoire.

- on s'intègre sans difficulté à un système moniteur. Pour les opérateurs ce sont des programmes Fortran normaux. Il suffit de placer derrière le programme les cartes binaires des fonctions FLPL. Cet avantage est très appréciable : on peut difficilement demander à un centre de calcul de modifier son système et la mise au point serait très ralentie si l'on devait passer hors moniteur.

- on dispose de toutes la puissance du Fortran pour les opérations qui ne portent pas sur des listes. Même quand on écrit des opérations de listes, le Fortran permet d'emboiter les fonctions ce qui donne une écriture très concise.

- il y a une très grande souplesse dans le stockage de l'information. Le mot de 36 bits est décomposés en 4 parties dont 2 de 15 bits qui peuvent servir de lien ou contenir l'information, une de 3 bits qui sert à indiquer si on a un lien vers une autre liste ou de l'information.

- rapidité. Il n'y a pas d'interprétation et la programmation des fonctions a été très bien faite.

On peut faire une critique assez mineure : la méthode d'effacement des listes inutiles. Il faut en effet donner un ordre spécial chaque fois que l'on veut effacer une sous liste. Dans certains cas

il est difficile de déterminer à partir de quand une liste devient inutile. Si on omet de l'effacer, on encombre la mémoire avec de l'information sans intérêt, si on donne deux fois l'ordre de l'effacer ou si on l'efface trop tôt, on a des erreurs de programme très difficiles à découvrir : le moment où l'erreur se manifeste est en général assez éloigné de celui où on l'a commise et le programme a eu le temps de supprimer la plupart des indices qui auraient pu servir à la localisation de l'erreur.

La solution adoptée par LISP parait très intéressante. On pourrait du reste l'adapter à FLPL. Il faudrait pour cela déclarer toutes les variables qui sont des noms de listes. Quand il n'y aurait plus de mémoires disponibles, on appellerait un sous programme qui mettrait le signe - dans le bit signe de toutes les mémoires auxquelles on peut parvenir en partant des listes dont on a déclaré le nom. Une mémoire pourrait être accessible de plusieurs façons sans que cela soit génant (à condition qu'il n'y ait pas de boucles). On retournerait à la liste des mémoires disponibles toutes les mémoires positives. On aurait la limitation de ne pas utiliser le bit signe, qui n'est pas génante pour le genre de problèmes que l'on a traité. J'aurais très certainement fait celle amélioration si j'avais écrit un cinquième programme.

La programmation en listes pose des problèmes originaux de mise au point. La méthode classique où l'on sort des résultats intermédiaires se révèle peu utile : dans la programmation classique, les erreurs viennent d'un résultat faux et la sortie de ce résultat rend l'erreur manifeste. Bien sûr, ce genre d'erreur se produit toujours, mais il s'y ajoute le cas où tous les résultats calculés sont exacts, mais où l'on a donné une mauvaise structure à l'information que l'on conserve. En conséquence l'erreur peut ne se manifester que longtemps après, au moment où l'on a besoin d'utiliser cette information mal mémorisée. La seule méthode efficace n'est donc pas de multiplier des sorties intermédiaires qui peuvent se révéler exactes, même en cas d'erreur. Il faut dumper la mémoire pour voir si elle est structurée convenablement. Cela risque de poser de graves problèmes quand on aura des machines dotées d'une capacité de mémoire beaucoup plus grande que celle des calculatrices actuelles. Une solution est peut être de concevoir un programme pouvant continuer de fonctionner même en cas d'erreur. L'utilisation de l'horloge et des mécanismes d'interruption peut donner à intervalles réguliers le contrôle à un programme qui vérifierait que l'on n'est pas dans un bouclage, que la mémoire a une structure correcte et ferait les corrections nécessaires. Tout ceci ne pose pas des problèmes de programmation très complexes en soi, mais risque d'amener des intéractions délicates avec le système.

L'outillage de listes est indispensable pour pouvoir utiliser des structures complexes dont la taille et les ramifications sont imprévisibles dans une large mesure et qui se modifient en cours d'exécution. Pour le programmeur, tout se passe comme s'il travaillait sur une machine fictive qui ferait les opérations dont il a besoin. Malheureusement les temps d'exécution sont élevés, car chaque opération sur les listes demande plusieurs instructions à la machine réelle.

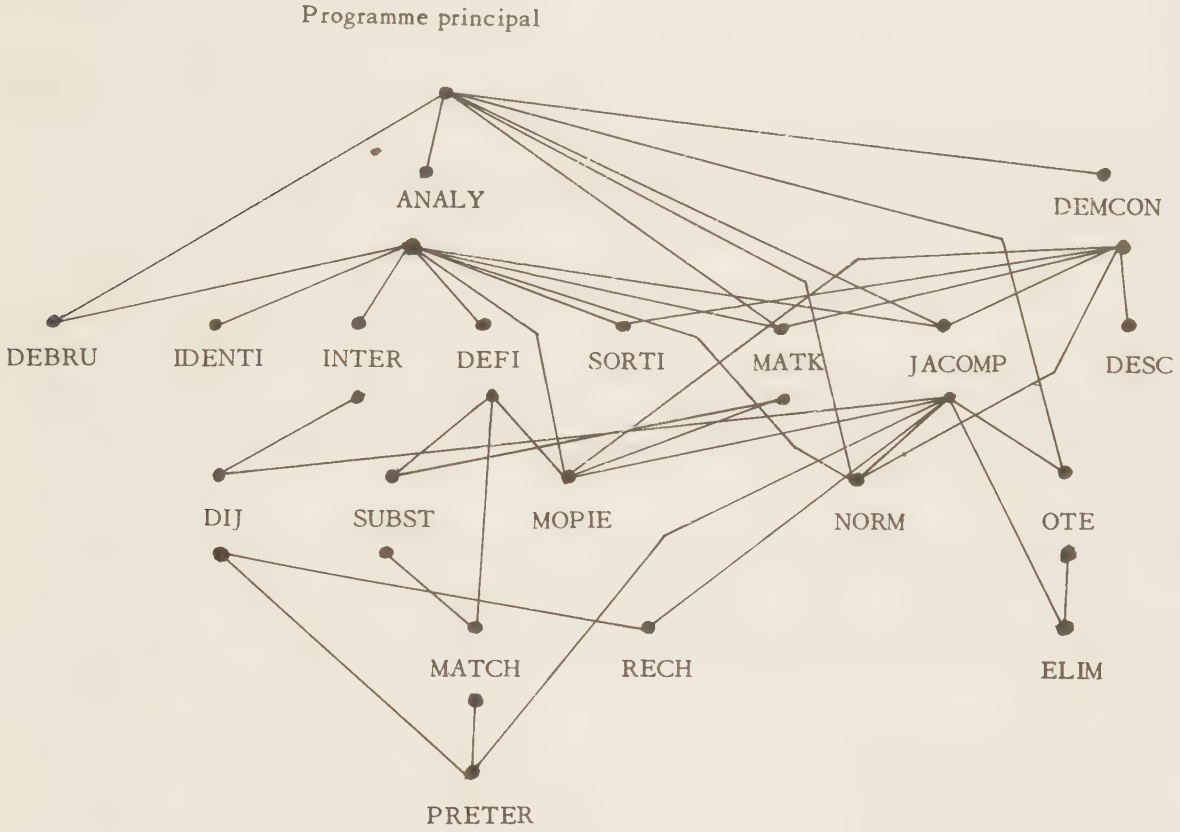
L'idéal serait de disposer d'une machine ayant les opérations de listes cablées. Il est paradoxal d'utiliser pour ce genre de travail des machines conçues pour le calcul scientifique : on n'utilise pas une seule fois la virgule flottante qui, elle, est cablée. L'intérêt de cabler les opérations de listes serait d'accélérer de façon considérable tous les programmes où on les utilise.

5C - LE QUATRIEME PROGRAMME

Il a été écrit et mis au point en 1964 et exploité en 1965 sur l'I.B.M. 704 de l'Institut Blaise Pascal. Il comporte 2200 instructions Fortran.

On peut représenter sa structure en liant par un trait un sous programme à tous les sous

programmes qu'il appelle, ces derniers étant placés à un niveau inférieur. On a ainsi un demi treillés qui a comme majorant universel le programme principal (en effet, il n'y a pas de récursivité en Fortran)



Remarque - Certains sous programmes ne figurent que dans un seul sous programme. Il se peut qu'ils y soient appelés plusieurs fois ou qu'ils constituent des unités logiques indépendantes qui permettent de segmenter des sous programmes qui auraient été trop volumineux.

Nous allons indiquer brièvement le rôle de ces divers sous programmes.

- Programme principal. Lit les données et créé les listes initiales. Sort en fin de travail les résultats archivés. Appelle ANALY qui va analyser les résultats, détermine quel va être l'essai suivant et appelle MATK qui fera l'essai. Il modifie les paramètres heuristiques susceptibles de s'adapter. Il appelle DEMCON s'il y a des conjectures à démontrer.

- ANALY. C'est de loin le sous programme le plus volumineux. Il calcule l'intérêt des résultats que l'on a trouvé. Si l'on a par exemple un théorème il va lui appliquer les MMthéorèmes que l'on connait et on aura des productions à un antécédent. Pour évaluer leur intérêt, il leur appliquera à nouveau des MMthéorèmes qui donneront naissance à des productions à deux antécédents dont on calculera l'intérêt. Cela permettra de connaitre celui des productions à un antécédent et finalement celui du théorème. Il regarde alors s'il y a lieu d'archiver le résultat. Si cela est possible, il créé de nouveaux MMthéorèmes.

- MATK. Essaye un résultat avec tous ceux qui figurent dans une arborescence. Si le résultat est un théorème et l'arborescence contient des productions à un antécédent, on aura de nouveaux théorèmes. Si elle contient des MMtheorèmes du type $a \longrightarrow b \Rightarrow c$, on aura de nouveaux Mthéorèmes qui sont des productions à un antécédent. Le type de déduction que l'on veut faire est indiqué par un des paramètres de la séquence d'appel (il y en a 16 au total).

- JACOMP. Regarde, en liaison avec ce que l'on a déjà trouvé, si le résultat est intéressant. Il exécute les opérations décrites en 3B3. Selon la nature du résultat, on fait une série de recherches décrites par des paramètres lus ce qui donne une très grande souplesse à ce programme.

- DEFI. Regarde s'il y a lieu d'appliquer une définition, et si oui, l'applique. Ce sont les opérations décrites en 4A.

- DEBRU. Il regarde si un résultat ne figure pas tel que dans une arborescence. S'il n'y est pas, il peut l'archiver.

- INTER. Calcule les valeurs des 21 paramètres qui sont utilisés pour l'évaluation de l'intérêt syntaxique d'une production à un antécédent (cf 3B2b).

- IDENTI. Cette fonction regarde si l'antécédent et le résultat d'une production ne sont pas identiques.

- SORTI. Il est indispensable d'avoir des sorties intermédiaires en plus de la sortie finale exécutée par le programme principal. En cas d'erreur de programme il faut pouvoir localiser l'erreur et surtout il faut suivre la suite des recherches du programme. On sort l'énoncé de tous les résultats trouvés, qu'ils soient ou non intéressants.

- DIJ. On étudie la force d'une production au sens de la première définition (3B1b) et on regarde s'il y a des parties neutres (3B2b).

- SUBST. Fait la dernière partie de l'opération de dérivation (cf 2B) en comparant la liste des termes à faire coïncider créée par MATK et en faisant les substitutions nécessaires.

- OTE. On balaye une arborescence en cherchant pour l'enlever un résultat dont on connait le nom ou tous les résultats dont l'intérêt est inférieur à un certain seuil.

- ELIM. On enlève un résultat d'une arborescence, sa position exacte étant indiquée par la séquence d'appel.

- NORM. Met les résultats sous forme canonique (cf remarque de 2A1c) en appelant p la première variable rencontrée, q la seconde---. Il peut également changer tous les noms des variables d'un résultat : on sera ainsi sûr qu'au moment d'un essai les variables des deux résultats qui interviennent seront différentes (cf 2B).

- MOPIE. Simple fonction de servitude qui recopie les listes. Cela est souvent utile quand on veut faire des substitutions dans un énoncé, et que l'on tient de garder l'original.

- MATCH. Sous programme secondaire lié à la dérivation.

- RECH. Sous programme secondaire utilisé pour étudier la force d'une production.

- PRETER. Détermine le terme (au sens de la notation polonaise) commençant en un point donné d'un énoncé.

- DEMCON. C'est lui qui démontre les conjectures. Il créé l'arbre "ou - et" et en assure l'élagage en cas d'échec ou de réussite partiels. Il fait les opérations décrites en 4D.

- DESC. Cherche les descendants d'une conjecture dans l'arbre "ou - et".

La mémoire de 32.768 mots a été utilisée de la façon suivante :

- système et sous programme auxiliaires (entrées-sorties, dump---)	3 600
- Instructions traduction du programme Fortran	14 000
- Fonctions FLPL	400
- Mémoire de travail non organisée en listes	2 000
- Mémoire organisée pour l'utilisation des listes	12 500

Remarque 1 - On voit la discrétion de l'outillage FLPL. Au cours de certaines exploitations où l'on n'essayait pas de démontrer des conjectures, on a gagné deux mille mots en supprimant DEMCON : on avait ainsi 14500 mots disponibles pour les listes. On a été constamment géné par le manque de place que l'on pouvait leur donner. Les listes ne sont pas utilisées dans de bonnes conditions et il aurait fallu pouvoir disposer pour elles d'au moins 100.000 mots. Quand il n'y a plus de mémoires libres, le programme doit s'arrêter.

Remarque 2 - Il est difficile de donner des évaluations de temps, car cela varie beaucoup d'une axiomatique à l'autre, en particulier avec la longueur moyenne des résultats (l'axiomatique 5 est très lente). Au cours de l'étude d'une axiomatique, le programme se ralentit au fur et à mesure qu'il a davantage de résultats, car cela augmente le nombre de recherches qu'il faut faire suivant la méthode exposée en 3B3.

A titre indicatif, les résultats donnés pour l'axiomatique 1 demandent 2 heures, la mémoire étant alors saturée. Avec l'axiomatique 4 qui a beaucoup moins de résultats, on a pu travailler 6 heures avant épuisement de la mémoire. Ces temps sont ceux sur 704. Les temps d'entrée sortie étant négligeables, il y a principalement des opérations de transfert et de test. On peut avoir facilement une idée des temps sur d'autres matériels : ils devraient être divisés par 5 pour un 7090 et par 10 pour un 7094 modèle 2.

Remarque 3 - La partie la plus volumineuse du programme est liée aux heuristiques. La partie déductive proprement dite est très réduite, grâce à la simplification apportée par l'utilisation de la notation polonaise. Elle n'a du reste posé aucun problème de mise au point : le programme a souvent sorti des résultats dénués d'intérêt, mais jamais des résultats qui n'étaient pas dérivables.

5D - ARCHIVAGE DES RESULTATS DANS DES ARBORESCENCES

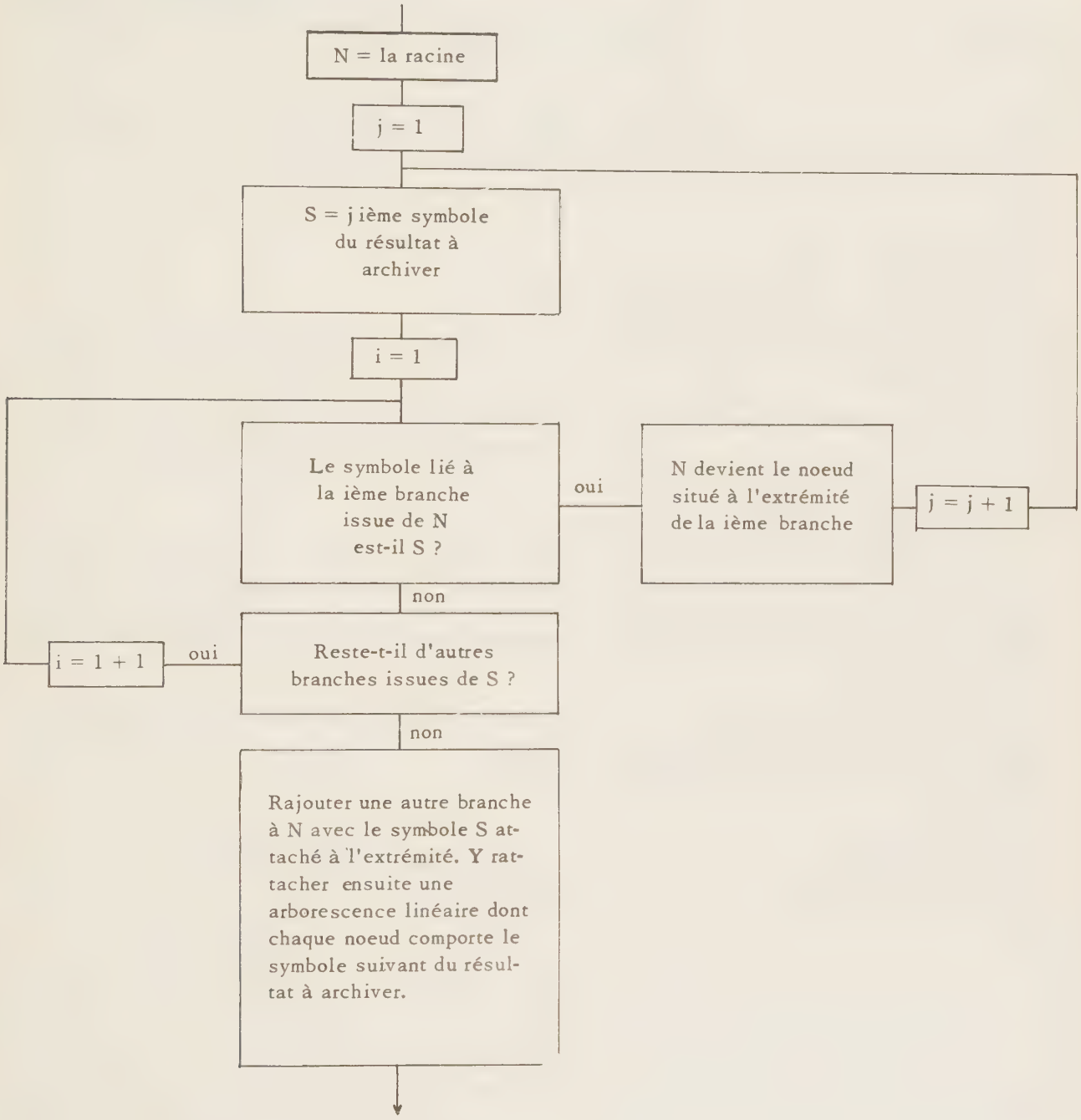
Cette méthode a été utilisée dans le but d'accélérer le programme. On peut trouver plus rapidement le résultat que l'on cherche et on peut faire certaines opérations en parallèle. Cela est surtout utile pour faire les recherches décrites en 3B3 et pour faire les dérivations.

A chaque noeud de l'arborescence est attachée une information : nom de connective ou nom de variable. On archive les résultats de telle façon qu'à tout chemin allant de la racine de l'arborescence à un sommet pendant, corresponde un résultat. Son énoncé s'obtient en écrivant de gauche à droite les symboles attachés au noeud que l'on rencontre quand on sait le chemin en partant de la racine. On ne met ni les Mconnectives, ni les MMconnectives. Il n'y aura pas d'ambiguité sur leur position, car nous verrons qu'il y a autant d'arborescences que de types de résultats. Dans une arborescence donnée, le programme sait exactement quelles Mconnectives ou MMconnectives il doit ajouter et où il doit les mettre. Cela n'aurait pas posé de difficulté de les faire figurer explicitement, mais on aurait perdu de la place inutilement.

Nous donnons un exemple de ce que peut être cette arborescence. On a archivé des théorèmes de l'axiomatique 1. A chaque noeud figure le symbole qui lui est attaché. Pour faciliter la lecture, on a ajouté à chaque sommet pendant l'énoncé du théorème correspondant au chemin qui se termine là.

Voyons comment s'effectuent les diverses opérations nécessaires sur ces arborescences. Nous donnerons des organigrammes qui ont le mérite d'être plus précis et plus clairs que les langues naturelles.

- ajouter un résultat.



Cet organigramme suppose que l'on est sûr que le résultat ne figure pas déjà dans l'arborescence. Il serait facile de le modifier pour tenir compte de cette éventualité. Quand on programme en listes, on n'a pas besoin d'avoir deux indices i et j : il suffit de prendre la branche suivante au lieu de faire i=i+1 et de prendre le symbole suivant de l'énoncé au lieu de faire j=j+1.

Si, dans l'exemple que l'on a donné, on voulait rajouter ⊃⊃pq⊃~~pq, il faudrait rajouter l'arborescence linéaire : "~pq" qprès le "~" qui est le sixième symbole de ⊃⊃pq⊃~q~p.

- éliminer un résultat.

Il faut n'enlever que la partie qui n'appartient qu'à lui. Sans cela on éliminerait aussi d'autres résultats. Il suffit de chercher le dernier noeud qui a plusiqurs branches dans le chemin qui lui correspond. On enlève toute la partie du chemin située derrière ce noeud.

Si l'on veut, dans l'exemple, éliminer ⊃⊃pq⊃⊃rp⊃rq, il faut enlever ce qui suit le quatrième ⊃ soit : "rp⊃rq". Si on enlevait ce qui suit le troisième ⊃, on enlèverait en même temps ⊃⊃pq⊃⊃qr⊃pr. Si on coupait plus bas, on n'aurait pas enlevé tout ce qui a trait à ⊃⊃pq⊃⊃rp⊃rq.

- balayer l'arborescence.

Les opérations de balayage sont fondamentales :

- pour lister les résultats en fin de travail.
- pour chercher un résultat ayant une certaine propriété (par exemple plus faible ou plus fort qu'un résultat donné)
- pour faire les dérivations avec un résultat d'une part et tous les résultats d'une arborescence d'autre part.
- pour éliminer des résultats dont on ne connait que le nom, ou les résultats dont l'intérêt est inférieur à une certaine valeur.

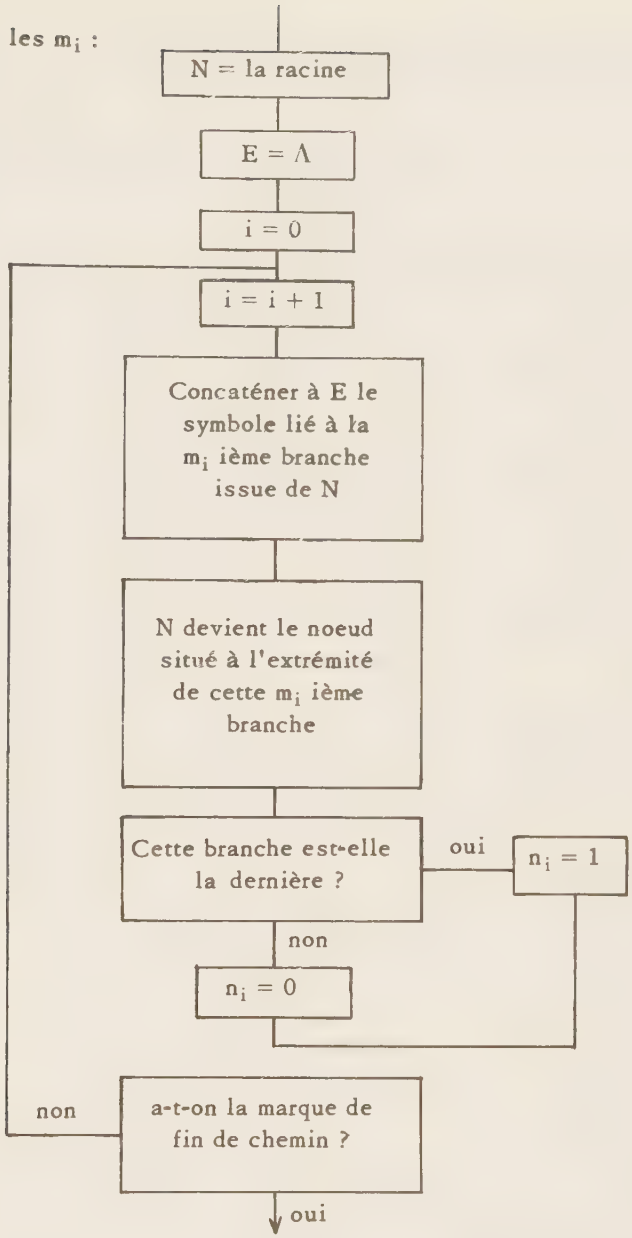
Pour cette opération on définit deux séries de grandeurs indicées : les m_i initialisés à 1 et les n_i dont la valeur initiale n'a pas d'importance. Les m_i vont indiquer le numéro de la branche de niveau i que l'on considère. Les n_i sont des variables booléennes qui indiquent s'il reste encore au niveau i des branches derrière celle qui fait partie du chemin considéré. Les niveaux sont comptés à partir de la racine qui est au niveau 0.

Il y a une marque spéciale mise au moment de l'archivage qui indique que l'on est arrivé à un sommet pendant. Ce n'est pas indispensable, car on aurait pu mettre un compteur de rang en notation polonaise, mais cela simplifie l'organigramme.

On décompose la recherche en deux phases :

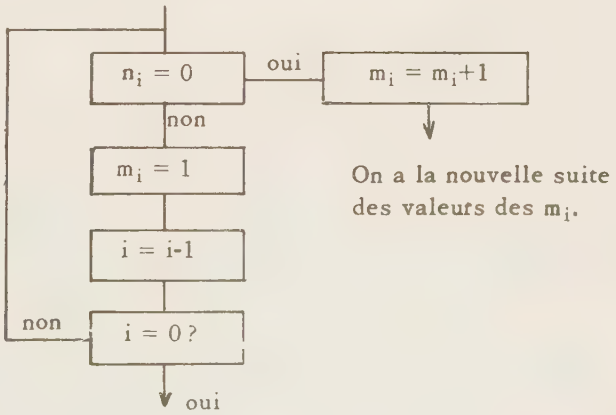
- recherche de l'énoncé du résultat correspondant à un certain ensemble de valeur des m_i.
- recherche des valeurs de m_i qui correspondent au résultat suivant.

Recherche de l'énoncé connaissant les m_i :



L'énoncé cherché sera E.

Recherche des valeurs de m_i correspondant au résultat suivant :



le balayage est terminé

La valeur initiale de i est celle qu'il avait quand, au cours de la recherche précédente, on est arrivé à un sommet pendant.

Le fait d'utiliser des variables indicées limite les dimensions des arborescences à celle donnée à ces variables. Pratiquement on a supposé que l'on avait jamais d'énoncé ayant plus de 40 symboles ce qui est largement suffisant. Les deux organigrammes se succèdent l'un à l'autre jusqu'à ce que l'arbre soit entièrement balayé.

Nous allons montrer comment on balaye l'arborescence donnée en exemple. On donne l'énoncé du théorème et les suites de valeurs de m_i et n_i qui lui correspondent en ne mettant que celles effectivement utilisées. On commence à m_1 et n_1.

Enoncé	valeur des m_i	valeur des n_i
C~pCpq	1 1 1 1 1 1	0 0 0 1 1 1
C~~pp	1 1 2 1 1	0 0 0 1 1
C~CpqCpr	1 1 3 1 1 1	0 0 1 1 1 1
Cpp	1 2 1	0 0 0
Cp~~p	1 2 2 1 1	0 0 0 1 1
CpCCpqq	1 2 3 1 1 1 1	0 0 1 0 1 1 1
CpCqp	1 2 3 2 1	0 0 1 0 1
Cp C~pq	1 2 3 3 1 1	0 0 1 1 1 1
CCpCqrCqCpr	1 3 1 1 1 1 1 1 1 1 1	0 0 0 0 1 1 1 0 1 1 1
CCpCqrCCpqCpr	1 3 1 1 1 1 1 2 1 1 1 1 1	0 0 0 0 1 1 1 1 1 1 1 1 1
CCpqC~q~p	1 3 1 2 1 1 1 1 1	0 0 0 0 1 0 1 1 1
CCpqCCqrCpr	1 3 1 2 1 2 1 1 1 1 1	0 0 0 0 1 1 0 1 1 1 1
CCpqCCrpCrq	1 3 1 2 1 2 2 1 1 1 1	0 0 0 0 1 1 1 1 1 1 1
CCp~qCq~p	1 3 1 3 1 1 1 1 1	0 0 0 1 1 1 1 1 1
CC~p~qCqp	1 3 2 1 1 1 1 1 1	0 0 1 1 0 1 1 1 1
CC~pqC~qp	1 3 2 1 2 1 1 1 1	0 0 1 1 0 1 1 1 1
CC~ppp	1 3 2 1 3 1	0 0 1 1 1 1
CApqp	1 4 1 1 1	0 0 1 1 0
CApqq	1 4 1 1 2	0 0 1 1 1
CEpqCpq	1 5 1 1 1 1 1	0 1 1 1 1 0 1
CEpqCqp	1 5 1 1 1 2 1	0 1 1 1 1 1 1
CEpp	2 1 1 1 1	1 1 1 1 1

Remarque 1 - On n'a pas introduit un ordre entre les différents symboles. Le dernier arrivé a la dernière branche. Les organigrammes n'auraient pas eu de grandes modifications, mais le gain aurait été minime : cela aurait permis d'arrêter la recherche dès que l'on arrive à un symbole supérieur au symbole cherché sans avoir besoin de tester toutes les branches. Bien entendu tous les énoncés sont mis sous forme canonique : le nom de la première variable qui apparait est p, celui de la seconde q---- pour éviter de multiplier les branches.

Remarque 2 - La methode indiquee est differente de celle proposée par IVERSON (A programming

language - Wiley and sons - 1962). Dans ses balayages d'arborescence, il suppose connu le nombre de branches issues de chaque noeud. Cela serait très compliqué de le garder en mémoire et de le tenir à jour. Il est plus simple de tester que l'on a balayé toutes les branches d'un noeud : on le fait par les n_i dont la valeur est calculée à l'aide du symbole qui indique la fin de la liste des branches du noeud étudié.

Remarque 3 - Il faut bien distinguer ces arborescences des arbres "ou - et" étudiés en 4D. Ici, à chaque chemin correspond un énoncé. Dans les arbres "ou - et", à chaque noeud correspond un énoncé, l'arbre établissant une hiérarchie entre les divers résultats.

Remarque 4 - On a utilisé les arborescences pour gagner du temps et non pour gagner de la place en mémoire. La place prise en appliquant cette méthode est très sensiblement la même que si l'on avait archivé les résultats les uns à la suite des autres. On y gagne d'un côté parce que certains symboles sont utilisés pour plusieurs résultats. Dans l'exemple que l'on a donné le premier $\supset$ est utilisé par 21 théorèmes et ne figure qu'une fois. Mais d'un autre côté on y perd, car il faut loger l'information qui décrit la structure de l'arborescence. Il faut (n+1) mémoires pour indiquer les n branches issues d'un noeud et le symbole qui y est attaché. Au total, bien que cela varie selon les axiomatiques, les deux méthodes sont équivalentes au point de vue occupation mémoire.

Le gain de temps est par contre considérable. Si l'on veut appliquer le Mthéorème $\supset\supset pqr \Rightarrow \supset qr$ à tous les résultats d'une arborescence, on voit tout de suite qu'on ne pourra l'appliquer à tous ceux qui commencent par $\sim$, $\supset\sim$, $\supset\wedge$, $\supset\equiv$. On peut même faire des opérations de dérivation simultanément. Si, toujours en appliquant le même Mthéorème on part de $\supset\supset pq\supset\sim q\sim p$ dans l'exemple que l'on a donné, en voit que l'on sera amené à faire les substitutions :

	P/p		P/p
	Q/q	puis	Q/q
	R/$\supset\sim q\sim p$		R/$\supset\supset qr\supset pr$
	P/p		P/p
puis	Q/q	puis	Q/$\sim q$
	R/$\supset\supset rp\supset rq$		R/$\supset q\sim p$

On voit que le premier couple est valable pour les 4 cas et le second pour les trois premiers. On a programmé le mécanisme de dérivation exposé en 2B d'une façon telle que l'on puisse utiliser ce fait.

Mais le gain est surtout important pour les recherches décrites en 3B3. On se ramène finalement toujours à chercher s'il existe dans une arborescence un résultat plus fort, égal ou plus faible qu'un résultat donné. Si l'on veut voir par exemple si $\supset\cdot p \supset p\sim qq$ n'est pas égal ou plus faible qu'un résultat de l'arborescence, quatre tests vont éliminer les 14 résultats qui ne commencent pas par $\supset\sim$ ou $\supset p$. Si l'on veut voir s'il n'est pas plus fort, 5 tests vont éliminer les 19 résultats qui ne commencent pas par $\supset\sim$.

Remarque 5 - Il y a au total 6 arborescences de résultats, les éléments de ces résultats étant rangés de façon à les utiliser de façon commode. Ce sont :

- Théorèmes.
- Productions à un antécédent $A \Rightarrow B$ rangées dans l'ordre A B. Ce sont les résultats qui sont intéressants à appliquer à un théorème et correspondent grosso modo aux productions fortes.

- Productions à un antécédent $A \Rightarrow B$ rangées dans l'ordre B A. Ce sont les résultats intéressants à appliquer à une conjecture. Elles correspondent à peu près aux productions faibles.

- Productions à deux antécédents $A,B \Rightarrow C$ rangées dans l'ordre C A B. Elle sont intéressantes à appliquer à une conjecture. On ne considère pas comme différentes $A,B \Rightarrow C$ et $B,A \Rightarrow C$.

- MMthéorèmes du type $A \longrightarrow B \Rightarrow C$ rangés dans l'ordre A B C. On les applique aux théorèmes pour en déduire des productions à un antécédent. On considère comme distincts $A \longrightarrow B \Rightarrow C$ et $B \longrightarrow A \Rightarrow C$ qui appliqué à un théorème donnent des résultats différents et on peut très bien les archiver tous les deux.

- MMthéorèmes du type $A \Rightarrow B \longrightarrow C, D \Rightarrow E$ rangés dans l'ordre A B C D E. Appliqués à une production à un antécédent, ils donnent une production à deux antécédents. On ne considère pas comme distincts $A \Rightarrow B \longrightarrow C, D \Rightarrow E$ et $A \Rightarrow B \longrightarrow D, C \Rightarrow E$: appliqués à la même production ils donnent le même résultat (à l'ordre des antécédents près).

CHAPITRE VI

RESULTATS OBTENUS PAR LE PROGRAMME

Nous exposerons de façon uniforme les résultats obtenus par le programme avec les diverses axiomatiques qu'il a étudiées : en tête nous placerons les axiomes, les productions et les définitions. Puis nous indiquerons les essais, en ne mettant que ceux qui ont donné, soit un résultat intéressant, soit un intermédiaire indispensable pour la suite. Chaque essai sera mis sous la forme :

a+b↦f c

a et b étant des numéros de résultats déjà trouvés, f l'énoncé du nouveau résultat et c son numéro. Le numéro sera formé de lettres suivies de chiffres ; la lettre sera T pour les théorèmes, M pour les Mthéorèmes et MM pour les MMthéorèmes. Les chiffres qui suivent sont des entiers pris selon l'ordre naturel, les numérotations de 3 catégories de résultats étant indépendantes. Nous indiquerons par

Tm+Di↦f Tn

que l'on a appliqué la définition Di au théorème Tm ce qui a donné le nouveau théorème Tn de formule f.

Nous n'indiquerons pas explicitement quel est le type de dérivation qui a été effectué, car il n'y a pratiquement jamais ambiguité :

T1+M4↦⊃p⊃⊃qrq⊃sq T3

signifie que l'on applique la production M4 au théorème T1 et qu'il en résulte le nouveau théorème T3 dont l'énoncé est ⊃p⊃⊃qrq⊃sq.

Le seul cas où il y ait ambiguité est celui où l'on applique une production à un antécédent à l'un des antécédents d'une production à 2 antécédents ; à priori on peut en effet l'appliquer à chacun des antécédents. Mais de tels essais sont rares, et l'énoncé du résultat permet de voir facilement quel est l'essai que l'on a effectué.

Dans certains cas, on peut faire figurer n'importe quel théorème. Par convention, nous mettrons alors X à la place du nom du théorème : Par exemple, dans l'axiomatique 3 où M5 est : p⇒⊃⊃⊃qrq⊃sq

X+M5↦⊃⊃⊃pqp⊃rp T4

Pour les MMthéorèmes, nous n'indiquerons même pas le MMMthéorème utilisé, celui-ci étant toujours évident. Nous indiquerons simplement : f MMi (f étant l'énoncé du MMthéorème et i son numéro), immédiatement au dessous de la démonstration du Mthéorème dont il est issu.

Les résultats les plus importants seront signalés par un astérisque placé à gauche de la ligne où ils sont trouvés.

Certains résultats ont été démontrés de diverses manières. Nous ne donnerons des variantes que si elles présentent un intérêt particulier.

Pour ne pas alourdir l'exposé, nous ne donnerons les formules de démonstrations que pour les premiers résultats des deux premières axiomatiques.

Citons trois ouvrages auxquels nous ferons fréquemment référence par la suite :

CHURCH : Introduction to mathematical logic. Volume 1. Princeton University Press.

DOPP : Leçons de logique formelle. Deuxième partie. Editions de l'Institut Supérieur de Philosophie-Louvain.

ROSENBLOOM : The elements of mathematical logic. Dover publications.

Axiomatique 1

Cette axiomatique est dûe à Lukasiewicz. Elle figure en particulier dans Church (qui l'appelle P2), dans Rosenbloom et dans Mendelson (Introduction to mathematical logic).

axiomes

$$\supset\supset p\supset qr\supset\supset pq\supset pr \quad T1$$

$$\supset p\supset qp \quad T2$$

$$\supset\supset\sim p\sim q\supset qp \quad T3$$

productions

$$p,\supset pq \Rightarrow q \quad M1$$

définitions

$$\wedge pq = \sim\supset p\sim q \quad D1$$

$$\equiv pq = \wedge\supset pq\supset qp \quad D2$$

* $\supset pq - \rightarrow p \Rightarrow q$ MM1

* $p \Rightarrow \supset qr \longrightarrow p,q \Rightarrow r$ MM2

T2+MM1 $\dashv\vdash p \Rightarrow \supset qp$ M3

T1+MM1 $\dashv\vdash \supset p\supset qr \Rightarrow \supset\supset pq\supset pr$ M4

M4+MM2 $\dashv\vdash \supset p\supset qr, \supset pq \Rightarrow \supset pr$ M5

* $\supset p\supset qr \longrightarrow \supset pq \Rightarrow \supset pr$ MM3

M3+M5 $\dashv\vdash \supset pq, \supset qr \Rightarrow \supset pr$ M6

* $\supset pq \longrightarrow \supset rp \Rightarrow \supset rq$ MM4

* $\supset pq \longrightarrow \supset qr \Rightarrow \supset pr$ MM5

T2+MM5 $\dashv\vdash \supset\supset pqr \Rightarrow \supset qr$ M7

* T1+M7 $\dashv\vdash \supset\supset pq\supset\supset rp\supset rq$ T4

M4+M7 $\dashv\vdash \supset p\supset qr \Rightarrow \supset q\supset pr$ M8

* T4+M8 $\dashv\vdash \supset\supset pq\supset\supset qr\supset pr$ T5

T4+MM1 $\dashv\vdash \supset pq \Rightarrow \supset\supset rp\supset rq$ M9

T5+MM1 $\dashv\vdash \supset pq \Rightarrow \supset\supset qr\supset pr$ M10

* T2+M10 $\dashv\vdash \supset\supset\supset pqr\supset qr$ T6

* T2+M8 $\dashv\vdash \supset\supset pq\supset p\supset rq$ T7

T6+MM4 $\dashv\vdash \supset p\supset\supset qrs \Rightarrow \supset p\supset rs$ M11

* T1+M11 $\dashv\vdash \supset\supset p\supset qr\supset q\supset pr$ T8

T8+MM5 $\dashv\vdash \supset\supset p\supset qrs \Rightarrow \supset\supset q\supset prs$ M12

T2+MM3 $\dashv\vdash \supset pq \Rightarrow \supset pp$ M13

* T2+M13 $\dashv\vdash \supset pp$ T9

* T9+M8 $\dashv\vdash \supset p\supset\supset pqq$ T10

* T10+M4 $\dashv\vdash \supset\supset p\supset pq\supset pq$ T11

T11+MM4 $\dashv\vdash \supset p\supset q\supset qr \Rightarrow \supset p\supset qr$ M14

* T3+M7⊣⊢⊃~p⊃pq T12

T12+MM5⊣⊢⊃⊃pqr⇒⊃~pr M15

M10+M15⊣⊢⊃pq⇒⊃~q⊃pr M16

T3+MM4⊣⊢⊃p⊃~q~r⇒⊃p⊃rq M17

M16+M17⊣⊢⊃~pq⇒⊃~q⊃rp M18

T11+MM1⊣⊢⊃p⊃pq⇒⊃pq M19

M18+M19⊣⊢⊃~pq⇒⊃~qp M20

* T9+M20⊣⊢⊃~~pp T13

T3+MM1⊣⊢⊃~p~q⇒⊃qp M21

* T13+M21⊣⊢⊃p~~p T14

T14+M9⊣⊢⊃⊃pq⊃p~~q T15

* T15+M17⊣⊢⊃⊃~pq⊃~qp T16

T16+M9⊣⊢⊃⊃p⊃~qr⊃p⊃~rq T17

T17+M14⊣⊢⊃⊃~p⊃~qp⊃~pq T18

T18+M12⊣⊢⊃⊃~p⊃~qq⊃~qp T19

T19+MM1⊣⊢⊃~p⊃~qq⇒⊃~qp M22

T13+M22⊣⊢⊃~p~⊃~pp T20

* T20+M21⊣⊢⊃⊃~ppp T21

T13+M10⊣⊢⊃⊃pq⊃~~pq T22

* T22+M17⊣⊢⊃⊃p~q⊃q~p T23

T15+MM5⊣⊢⊃⊃p~~qr⇒⊃⊃pqr M23

* T23+M23⊣⊢⊃⊃pq⊃~q~p T24

T24+MM1⊣⊢⊃pq⇒⊃~q~p M24

T14+D1⊣⊢⊃⊃p~q~∧pq T25

T25+MM1⊣⊢⊃p~q⇒~∧pq M25

* T14+M25⊣⊢~∧p~p T26

T2+M20⊣⊢⊃~⊃p~qq T27

* T27+D1⊣⊢⊃∧pqq T28

* T28+D2⊣⊢⊃≡pq⊃qp T29

T29+MM1⊣⊢≡pq⇒⊃qp M26

M26+MM2⊣⊢p,≡qp⇒q M27

* ≡pq⟶q⇒p MM6

* T12+M20⊣⊢⊃~⊃pqp T30

* T30+D1⊣⊢⊃∧pqp T31

* T31+D2⊣⊢⊃≡pq⊃pq T32

T32+MM1⊣⊢≡pq⇒⊃pq M28

M28+MM2⊣⊢p,≡pq⇒q M29

* ≡pq⟶p⇒q MM7

* T32+M8⊣⊢⊃p⊃≡pqq T33

* T29+M8⊣⊢⊃p⊃≡qpq T34

* T33+M4⊣⊢⊃⊃p≡pq⊃pq T35

* T12+M8⊣⊢⊃p⊃~pq T36

* T36+M10⊣⊢⊃⊃⊃~pqr⊃pr T37

T22+MM5⊣⊢⊃⊃~~pqr⇒⊃⊃pqr M30

* T21+M30⊣⊢⊃⊃p~p~p T38

T23+M9⊣⊢⊃⊃p⊃q~r⊃p⊃r~q T39

T39+M14⊣⊢⊃⊃p⊃q~p⊃p~q T40

T40-MM1⊣⊢⊃p⊃q~p⇒⊃p~q M31

T10+M31⊣⊢⊃p~⊃p~p T41

* T41+D1⊣⊢⊃p∧pp T42

Nous allons maintenant démontrer le Mthéorème ⊃~p⊃qp⇒⊃qp qui va permettre de démontrer plusieurs résultats intéressants et une variante de la démonstration de T21 qui a pour particularité d'utiliser le Mtheorème ⊃~pp⇒p. La démonstration du programme comportant des maladresses, nous y ferons exceptionnellement deux modifications que nous indiquerons.

T12+MM3⊣⊢⊃~pp⇒⊃~pq M32 (en réalité le programme est parti de ⊃~p⊃p~q)

M32+M21⊣⊢⊃~pp⇒⊃qp M33

M33+MM2⊣⊢⊃~pp,q⇒p M34

X+M34⊣⊢⊃~pp⇒p M35

T2+M24⊣⊢⊃~⊃pq~q T43 (en réalité ce théorème a été obtenu d'une autre façon)

T43+MM5⊣⊢⊃~pq⇒⊃~⊃rpq M36

M36+M35⊣⊢⊃~p⊃qp⇒⊃qp M37

et l'on a :

* T10+M37⊣⊢⊃⊃~ppp T21

* T33+M37⊣⊢⊃≡~ppp T44

* T34+M37⊣⊢⊃≡p~pp T45

Citons une autre variante que j'ai trouvé pour T21 et qui est indépendante des deux démonstrations données par le programme :

T24+MM4⊣⊢⊃p⊃qr⇒⊃p⊃~r~q M38

T10+M38⊣⊢⊃p⊃~q~⊃pq T46

T46+M19⊣⊢⊃~p~⊃~pp T20

T20+M21⊣⊢⊃⊃~ppp T21

Montrons comment l'on peut obtenir les formules de démonstration correspondant aux premiers résultats trouvés. Nous donnerons le nom du résultat suivi de sa formule. Nous expliciterons les productions à 2 antécédents pour qu'il n'y ait pas d'ambiguité : le premier antécédent correspondra à i et le deuxième à j.

T1		a	
T2		b	
T3		c	
M1		Rij	$p, \supset pq \Rightarrow q$
MM1		$\alpha \longrightarrow R i \alpha$	
MM2		$\alpha i \beta \longrightarrow R j \alpha i \beta$	
M3		Rib	
M4		Ria	
M5		RjRia	$\supset p \supset qr, \supset pq \Rightarrow \supset pr$
MM3		$\alpha \longrightarrow R i R \alpha a$	
M6		RjRRiba	$\supset pq, \supset rp \Rightarrow \supset rq$
	ou	RiRRjba	$\supset pq, \supset qr \Rightarrow \supset pr$
MM4		$\alpha \longrightarrow R i R R \alpha b a$	
MM5		$\alpha \longrightarrow R \alpha R R i b a$	
M7		RbRRiba	
T4		RbRRaba	
M8		RbRRRiaba	
T5		RbRRRRbRRabaaba	
M9		RiRbRRaba	
M10		RiRbRRRRbRRabaaba	
T6		RbRbRRRRbRRabaaba	
M11		RiRRRbRbRRRRbRRabaababa	
T8		RaRRRbRbRRRRbRRabaababa	
M13		RiRba	
T9		RbRba	
T10		RbRRRRbRbaaba	
T11		RRbRRRRbRbaabaa	

On pourrait continuer sans difficulté. Les formules devenant de plus en plus lourdes, il est commode d'introduire des définitions du type :

d pour RbRba

ou Mi pour RbRRRiaba

ou Sij pour RjRRiba

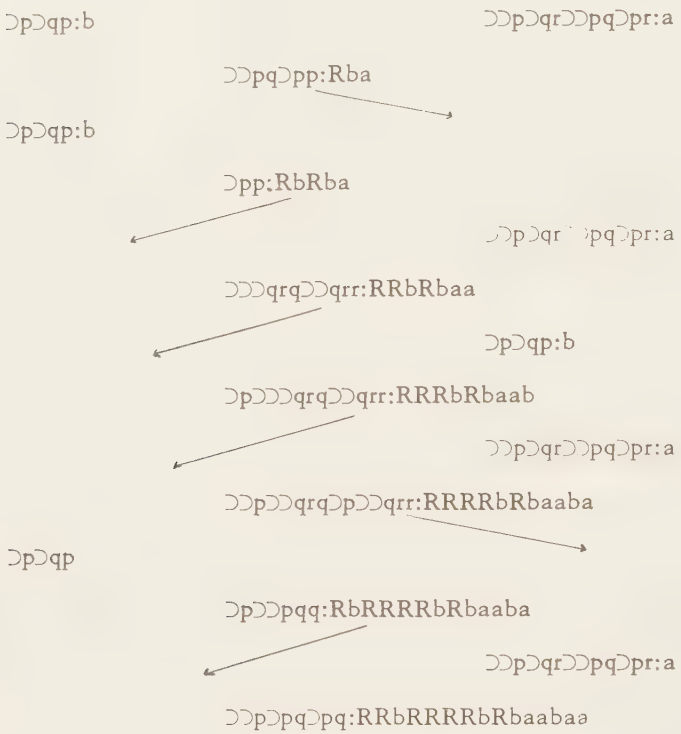
T10 deviendrait alors Md ce qui ne ferait que traduire :

$$\supset pp + \supset p \supset qr \Rightarrow \supset q \supset pr \dashrightarrow \supset p \supset \supset pqq$$

Signalons que pour M9 il existe une formule de démonstration plus simple : RRiba.

A partir de la formule de démonstration de T11 nous allons montrer comment on peut présenter

la même dérivation, sans faire appel aux Mthéorèmes. Nous prendrons pour cela un procédé exposé en 2C2. Le théorème substitué au premier antécédent de M1 figurera à gauche, au deuxième à droite. Au dessous on indiquera l'énoncé du résultat suivi de sa formule de démonstration. S'il est utilisé dans la dérivation suivante, il ne sera pas répété, mais une flèche indiquera la place où il doit être.



Remarquons au passage que l'on a montré que tout théorème de l'axiomatique 2 est un théorème de l'axiomatique 1. En effet, ces deux axiomatiques ont la même production et les axiomes de l'axiomatique 2 sont des théorèmes de l'axiomatique 1 : T21, T36 et T5. On a donc démontré également tous les théorèmes de cette axiomatique, et en particulier CCCpqpp.

Comparons les résultats obtenus par le programme avec l'exposé de Rosenbloom. Le programme a démontré 15 des théorèmes donnés par Rosenbloom (dont l'un, M29, est en réalité un Mthéorème). Le programme a trouvé des théorèmes que Rosenbloom ne cite pas, par exemple T6, T11, T21, T30, T36,T42--- Par contre il n'a pas trouvé plusieurs résultats donnés par Rosenbloom.

Les démonstrations, et même l'ordre dans lequel les théorèmes sont trouvés sont souvent très différentes : Rosenbloom démontre T8 pour en déduire M8 et démontrer ensuite T5. En réalité on peut obtenir M8 indépendamment de T8. La démonstration qu'il donne pour ce dernier est très peu intuitive. Il l'a obtenue probablement en appliquant le théorème de déduction à M8. La démonstration du programme, qui fait disparaitre le deuxième p de T1 est beaucoup plus élégante.

AXIOMATIQUE 2

Cette axiomatique est dûe à Lukasiewicz qui l'étudie en détail dans les "Elements of mathematical logic" - Pergamon Press. Elle est citée par Church qui la propose dans l'exercice 29-2.

axiomes

$CC\sim ppp$ T1

$Cp C\sim pq$ T2

$CCpqCCqrCpr$ T3

production

$p, Cpq \Rightarrow q$ M1

* $Cpq \dashrightarrow p \Rightarrow q$ MM1

* $p \Rightarrow Cqr \longrightarrow p, q \Rightarrow r$ MM2

T3+MM1 $\dashrightarrow Cpq \Rightarrow CCqrCpr$ M2

M2+MM2 $\dashrightarrow Cpq, Cqr \Rightarrow Cpr$ M3

$Cpq \longrightarrow Crp \Rightarrow Crq$ MM3

* $Cpq \longrightarrow Cqr \Rightarrow Cpr$ MM4

T3+MM4 $\dashrightarrow CCCpqCrqs \Rightarrow CCrps$ M4

T2+M5 $\dashrightarrow CC\sim pqr \Rightarrow Cpr$ M5

M2+M5 $\dashrightarrow Cp\sim q \Rightarrow Cq Cpr$ M6

T2+MM1 $\dashrightarrow p \Rightarrow C\sim pq$ M7

M7+M6 $\dashrightarrow p \Rightarrow Cq C\sim pr$ M8

T1+MM3 $\dashrightarrow Cp C\sim qq \Rightarrow Cpq$ M9

M2+M9 $\dashrightarrow C\sim pq \Rightarrow CCqpp$ M10

M8+M9 $\dashrightarrow p \Rightarrow Cqp$ M11

M11+M10 $\dashrightarrow p \Rightarrow CCpqq$ M12

* M12+M4 $\dashrightarrow Cpq \Rightarrow CCrpCrq$ M13

M6+M9 $\dashrightarrow C\sim p\sim q \Rightarrow Cqp$ M14

* T2+M2 $\dashrightarrow CCC\sim pqrCpr$ T4

T4+MM4 $\dashrightarrow CCpqr \Rightarrow CCC\sim psqr$ M15

M15+M4 $\dashrightarrow CCpCqrs \Rightarrow CCq\sim ps$ M16

T1+M13 $\dashrightarrow CCpC\sim qqCpq$ T5

* T5+M16 $\dashrightarrow CC\sim p\sim qCqp$ T6

* T6+M5 $\dashrightarrow CpCqp$ T7

T6+MM3 $\dashrightarrow CpC\sim q\sim r \Rightarrow CpCrq$ M17

* T7+M17 $\dashrightarrow C\sim pCpq$ T8

* T8+M9 $\dashrightarrow C\sim\sim pp$ T9

* T9+M14 $\dashrightarrow Cp\sim\sim p$ T10

* T2+M9 $\dashrightarrow Cpp$ T11

* T7+M13 $\dashrightarrow CCpqCpCrq$ T12

* T8+M10 $\dashrightarrow CCCpqpp$ T13

* T7+M2 $\dashrightarrow CCCpqrCqr$ T14

T14+MM1 $\dashrightarrow CCpqr \Rightarrow Cqr$ M18

T5+M4 $\vdash\!\!\rightarrow$ ⊃⊃~pq⊃⊃qpp T15

* T15+M18 $\vdash\!\!\rightarrow$ ⊃p⊃⊃pqq T16

T10+M13 $\vdash\!\!\rightarrow$ ⊃⊃pq⊃p~~q T17

* T17+M17 $\vdash\!\!\rightarrow$ ⊃⊃~pq⊃~qp T18

T9+M2 $\vdash\!\!\rightarrow$ ⊃⊃pq⊃~~pq T19

T19+MM4 $\vdash\!\!\rightarrow$ ⊃⊃~~pqr⇒⊃⊃pqr M19

* T18+M19 $\vdash\!\!\rightarrow$ ⊃⊃pq⊃~q~p T20

T10+M2 $\vdash\!\!\rightarrow$ ⊃⊃~~pq⊃pq T21

T21+MM3 $\vdash\!\!\rightarrow$ ⊃p⊃~~qr⇒⊃p⊃qr M20

* T20+M20 $\vdash\!\!\rightarrow$ ⊃⊃p~q⊃q~p T22

* T9+M10 $\vdash\!\!\rightarrow$ ⊃⊃p~p~p T23

T22+MM1 $\vdash\!\!\rightarrow$ ⊃p~q⇒⊃q~p M21

* T23+M21 $\vdash\!\!\rightarrow$ ⊃p~⊃p~p T24

si l'on avait la définition du "et", ce serait ⊃p/\pp

T6+M2 $\vdash\!\!\rightarrow$ ⊃⊃⊃pqr⊃⊃~q~pr T25

T25+MM4 $\vdash\!\!\rightarrow$ ⊃⊃⊃~p~qrs⇒⊃⊃⊃qprs M22

* T14+M22 $\vdash\!\!\rightarrow$ ⊃⊃⊃pqr⊃~pr M26

T26+MM1 $\vdash\!\!\rightarrow$ ⊃⊃pqr⇒⊃~pr M23

T14+M23 $\vdash\!\!\rightarrow$ ⊃~⊃pq⊃qr T27

T1+MM1 $\vdash\!\!\rightarrow$ ⊃~pp⇒p M24

T2+MM3 $\vdash\!\!\rightarrow$ ⊃pq⇒⊃⊃p⊃~qr M25

M25+M24 $\vdash\!\!\rightarrow$ ⊃~⊃~pqp⇒⊃~pq M26

* T27+M26 $\vdash\!\!\rightarrow$ ⊃~⊃pqp T28

* T28+M10 $\vdash\!\!\rightarrow$ ⊃⊃p⊃pq⊃pq T29

Donnons quelques variantes intéressantes de certaines démonstrations :

T15+MM3 $\vdash\!\!\rightarrow$ ⊃p⊃~qr⇒⊃p⊃⊃rqq M27

T7+M27 $\vdash\!\!\rightarrow$ ⊃p⊃⊃pqq T16

T4+MM3 $\vdash\!\!\rightarrow$ ⊃p⊃⊃~qrs⇒⊃p⊃qs M28

T16+M28 $\vdash\!\!\rightarrow$ ⊃~p⊃pq T8

En prenant la même disposition que pour l'axiomatique 1, donnons les formules de démonstration des premiers résultats.

T1	a
T2	b
T3	c

M1		Rij	$p, \supset pq \Rightarrow q$
MM1		$\alpha \longrightarrow R i \alpha$	
MM2		$\alpha i \beta \longrightarrow R j \alpha i \beta$	
M2		Ric	
M3		RjRic	$\supset pq, \supset qr \Rightarrow \supset pr$
	ou	RiRjc	$\supset pq, \supset rp \Rightarrow \supset rq$
MM3		$\alpha \longrightarrow R \alpha R i c$	
MM4		$\alpha \longrightarrow R i R \alpha c$	
M4		RiRcc	
M5		RiRbc	
M6		RRicRbc	
M7		Rib	
M8		RRRibcRbc	
M9		RaRic	
M10		RaRRicc	
M11		RaRRRRibcRbcc	
M12		RaRRRaRRRRibcRbccccRcc	
T4		Rbc	
M15		RiRRbcc	
M16		RRiRRbccRcc	
T5		RRaRRRaRRRRabcRbccccRcc	
T6		RRRRaRRRaRRRRabcRbccccRccRRbccRcc	
T7		RRRRRaRRRaRRRRabcRbccccRccRRbccRccRbc	

On voit que cette démonstration de T7 a pour profondeur 13 et pour taille 19.

A partir de la formule de T7, on va donner ce que serait la dérivation si l'on ne faisait pas appel aux Mthéorèmes. Pour simplifier nous donnerons des noms à certains résultats intermédiaires.

$\supset p \supset \sim pq$ $\qquad\qquad$ $\supset\supset pq \supset\supset qr \supset pr$

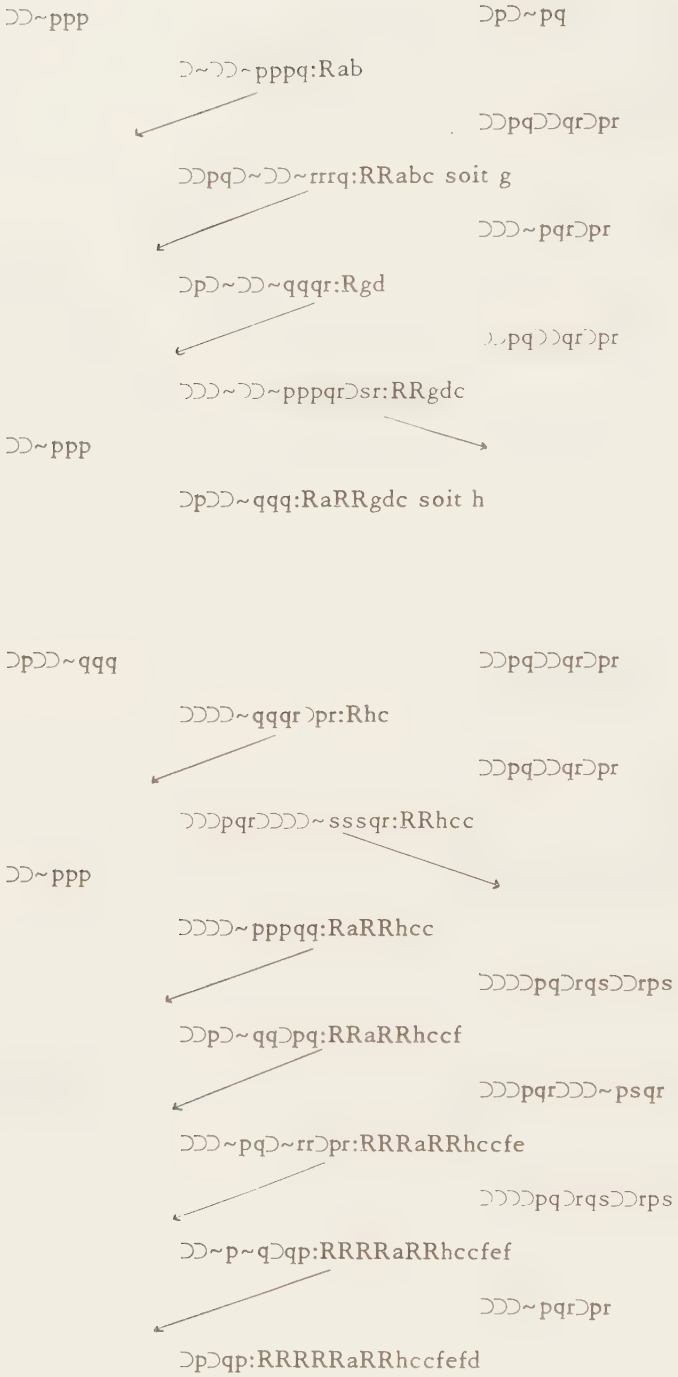
$\supset\supset\supset \sim pqr \supset pr$: Rbc soit d

$\supset\supset pq \supset\supset qr \supset pr$

$\supset\supset\supset pqr \supset\supset\supset \sim psqr$: RRbcc ou Rdc soit e

$\supset\supset pq \supset\supset qr \supset pr$ $\qquad\qquad$ $\supset\supset pq \supset\supset qr \supset pr$

$\supset\supset\supset\supset pq \supset rqs \supset\supset rps$: Rcc soit f



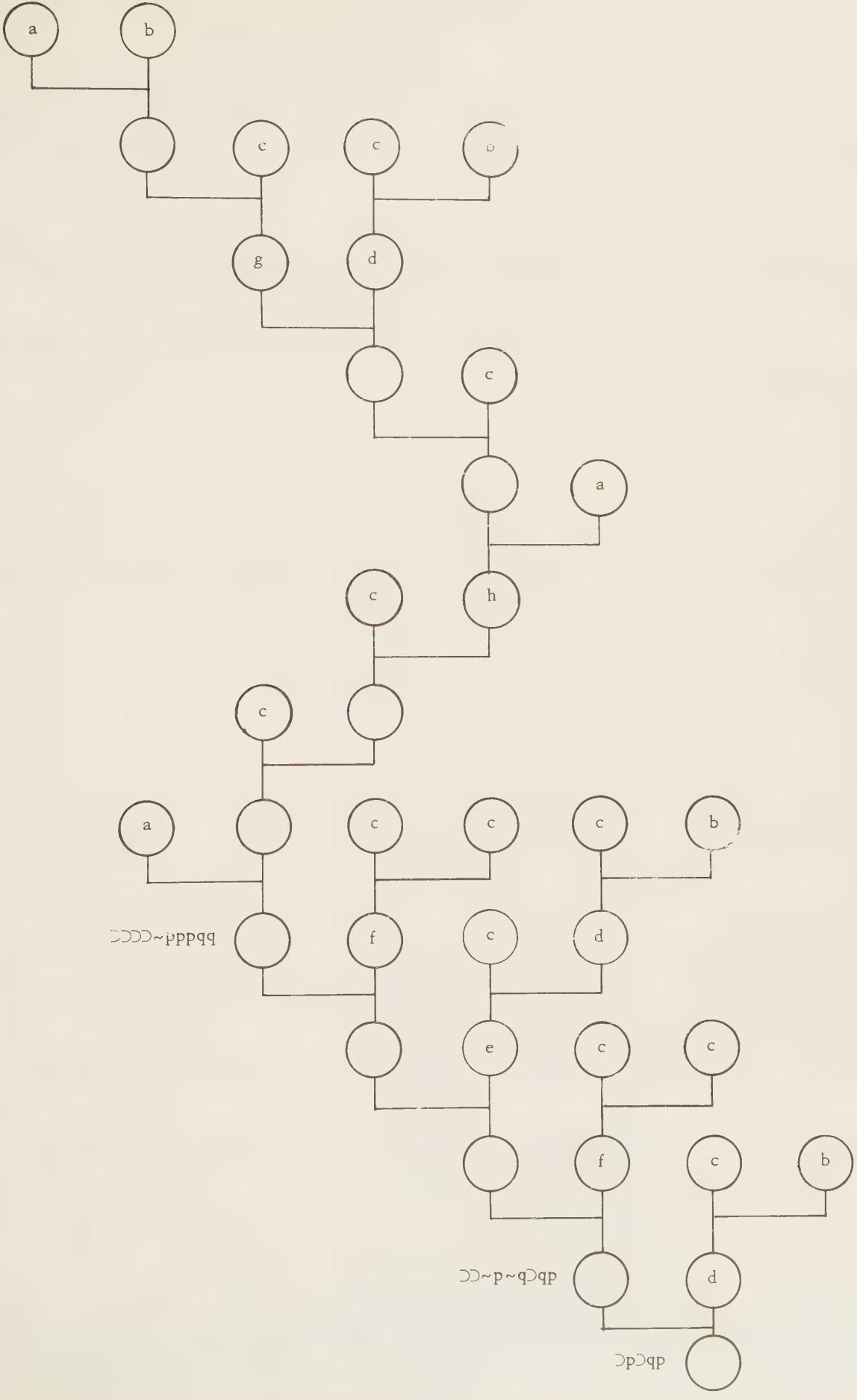
Nous allons également pour cette démonstration donner sa représentation graphique.

Remarquons que T3, T7 et T13 sont les trois axiomes de l'axiomatique 4 qui a pour seule production M1. Tout théorème de l'axiomatique 4 est donc un théorème de l'axiomatique 2 et en particulier on a :

CCpCqrCqCpr T30

CCpqCCrpCrq T31

CCpCqrCCpqCpr T32


a
b
c
c
g
d
c
a
c
h
c
a
c
c
c
b
f
c
d
e
c
c
f
c
b
d

les démonstrations données dans l'axiomatique 4 étant entièrement valables.

Mais alors on a prouvé l'équivalence des axiomatiques 1 et 2. On a déjà vu que les 3 axiomes de l'axiomatique 2 étaient des théorèmes de l'axiomatique 1. Inversement les 3 axiomes de l'axiomatique 1 sont les théorèmes T32, T7 et T6 de l'axiomatique 2. Tout théorème de l'une est théorème de l'autre.

La marche suivie par le programme est différente à et ce n'est que quand il a établi les Mthéorémes importants qu'il démontre les théorèmes. Il démontre M11, M12, M13, M14 bien avant les théorèmes correspondants et souvent le Mthéorème est indispensable pour démontrer le théorème.

La marche suivi par le programme est différente à la fois de celle donnée par Lukasiewicz et de celle proposée par Church. C'est ainsi qu'il déduit très simplement T7 de T6 alors que Lukasiewicz et Church démontrent directement T7.

AXIOMATIQUE 3

Cette axiomatique est dûe à Lukasiewicz qui l'étudie dans "The shortest axiom of the implicational calculus of propositions". Proceedings of the Royal Irish Academy - 1948 - Vol 52 - Section A N° 3 p. 25. Elle constitue l'exercice 18-4 de Church.

axiome $\supset\supset\supset pqr\supset\supset rp\supset sp$ T1

production $p,\supset pq \Rightarrow q$ M1

* $\supset pq \longrightarrow p \Rightarrow q$ MM1

* $p \Rightarrow \supset qr \longrightarrow p,q \Rightarrow r$ MM2

T1+MM1 $\dashv\vdash \supset\supset pqr \Rightarrow \supset\supset rp\supset sp$ M2

M2+MM2 $\dashv\vdash \supset\supset pqr,\supset rp \Rightarrow \supset sp$ M3

* $\supset\supset pqr \longrightarrow \supset rp \Rightarrow \supset sp$ MM3

T1+M2 $\dashv\vdash \supset\supset\supset pq\supset rq\supset qs\supset t\supset qs$ T2

T2+MM3 $\dashv\vdash \supset\supset p\supset qr\supset\supset sq\supset tq \Rightarrow \supset u\supset\supset sq\supset tq$ M4

T1+M4 $\dashv\vdash \supset p\supset\supset\supset qrq\supset sq$ T3

T3+MM1 $\dashv\vdash p \Rightarrow \supset\supset\supset qrq\supset sq$ M5

* X+M5 $\dashv\vdash \supset\supset\supset pqp\supset rp$ T4

T4+MM1 $\dashv\vdash \supset\supset\supset pqp \Rightarrow \supset rp$ M6

M6+MM2 $\dashv\vdash \supset\supset\supset pqp,r \Rightarrow p$ M7

X+M7 $\dashv\vdash \supset\supset\supset pqp \Rightarrow p$ M8

* T4+M8 $\dashv\vdash \supset pp$ T5

T1+MM3 $\dashv\vdash \supset\supset\supset pq\supset rq\supset qs \Rightarrow \supset t\supset qs$ M9

M9+MM2 $\dashv\vdash \supset\supset\supset pq\supset rq\supset qs,t \Rightarrow \supset qs$ M10

X+M10 ⊢→ ⊃⊃⊃pq⊃rq⊃qs⇒⊃ps M11

M2+M11 ⊢→ ⊃⊃⊃pqr⊃sq⇒⊃q⊃pq M12

* T5+M12 ⊢→ ⊃p⊃qp T6

T4+M2 ⊢→ ⊃⊃⊃pq⊃qr⊃s⊃qr T7

T7+M2 ⊢→ ⊃⊃⊃p⊃qr⊃sq⊃t⊃sq T8

T8+MM3 ⊢→ ⊃⊃p⊃qr⊃s⊃rt⇒⊃u⊃s⊃rt M13

M13+MM2 ⊢→ ⊃⊃p⊃qr⊃s⊃rt,u⇒⊃s⊃rt M14

X+M14 ⊢→ ⊃⊃p⊃qr⊃s⊃rt⇒⊃s⊃rt M15

* T1+M15 ⊢→ ⊃⊃⊃pqr⊃qr T9

T7+MM3 ⊢→ ⊃⊃p⊃qr⊃sq⇒⊃t⊃sq M16

M16+MM2 ⊢→ ⊃⊃p⊃qr⊃sq,t⇒⊃sq M17

X+M17 ⊢→ ⊃⊃p⊃qr⊃sq⇒⊃sq M18

T1+M18 ⊢→ ⊃⊃⊃⊃pqrq⊃pq T10

T10+MM3 ⊢→ ⊃⊃pq⊃⊃pqr⇒⊃s⊃⊃pqr M19

M19+MM2 ⊢→ ⊃⊃pq⊃⊃pqr,s⇒⊃⊃pqr M20

X+M20 ⊢→ ⊃⊃pq⊃⊃pqr⇒⊃⊃pqr M21

* T4+M21 ⊢→ ⊃⊃⊃pqpp T11

M2+M21 ⊢→ ⊃⊃pqr⇒⊃⊃rpp M22

* T11+M22 ⊢→ ⊃⊃p⊃pq⊃pq T12

Lukasiewicz montre que le calcul des propositions basé sur l'implication peut être dérivé de cet axiome unique et qu'il n'existe pas d'axiome plus court ayant cette propriété. Pour démontrer que l'on peut en déduire le calcul implicationnel, il démontre les trois axiomes de l'axiomatique 4.

Le programme a démontré les deux premiers (T6 et T11). La démonstration qu'il donne pour ⊃⊃⊃pqpp est très différente de celle de Lukasiewicz et beaucoup plus directe. Par contre le programme n'a pu démontrer le troisième axiome : ⊃⊃pq⊃⊃qr⊃pr ; il a seulement trouvé un intermédiaire utile : ⊃⊃⊃pqr⊃qr, en donnant la même démonstration que Lukasiewicz (à l'époque où le programme l'a trouvé, je ne connaissais pas l'article de Lukasiewicz et je n'avais pas réussi à le démontrer moi-même). La démonstration de ⊃⊃pq⊃⊃qr⊃pr est très compliquée et dépasse les possibilités du programme. On ne voit pas du reste comment Lukasiewicz a pu arriver à trouver cette démonstration extrémement complexe. Il a été très certainement aidé par ce qui l'a, en sens inverse, amené à considérer ⊃⊃⊃pqr⊃⊃rp⊃sp comme axiome.

Les étapes proposées par Church sont celles trouvées par Lukasiewicz

AXIOMATIQUE 4

Cette axiomatique est dûe, d'après Dopp, à Hilbert et Bernays. Elle est citée par Church qui la propose dans les exercices 12-7 et 12-8. Elle est également citée par Lukasiewicz qui montre qu'elle est équivalente à l'axiomatique 3. Cette formulation "positive" du calcul des propositions ne fait intervenir qu'une seule connective : l'implication.

axiomes

$CCpqCCqrCpr$ T1

$CpCqp$ T2

$CCCpqpp$ T3

production

$p, Cpq \Rightarrow q$ M1

* $Cpq \longrightarrow p \Rightarrow q$ MM1

* $p \Rightarrow Cqr \longrightarrow p, q \Rightarrow r$ MM2

T1+MM1 $\dashrightarrow Cpq \Rightarrow CCqrCpr$ M2

M2+MM2 $\dashrightarrow Cpq, Cqr \Rightarrow Cpr$ M3

* $Cpq \longrightarrow Crp \Rightarrow Crq$ MM3

* $Cpq \longrightarrow Cqr \Rightarrow Cpr$ MM4

T1+MM4 $\dashrightarrow CCCpqCrqs \Rightarrow CCrps$ M4

M2+M4 $\dashrightarrow CpCqr \Rightarrow CCsqCpCsr$ M5

T3+MM3 $\dashrightarrow CpCCqrq \Rightarrow Cpq$ M6

* T1+M6 $\dashrightarrow CCpCpqCpq$ T4

T4+MM1 $\dashrightarrow CpCpq \Rightarrow Cpq$ M7

M5+M7 $\dashrightarrow CCpqCqr \Rightarrow CCpqCpr$ M8

T2+MM1 $\dashrightarrow p \Rightarrow Cqp$ M9

* M9+M8 $\dashrightarrow Cpq \Rightarrow CCrpCrq$ M10

T4+MM3 $\dashrightarrow CpCqCqr \Rightarrow CpCqr$ M11

M10+M11 $\dashrightarrow CpCqr \Rightarrow CCqpCqr$ M12

T4+MM4 $\dashrightarrow CCpqr \Rightarrow CCpCpqr$ M13

T2+M13 $\dashrightarrow CCpCpqCrCpq$ T5

T5+M12 $\dashrightarrow CCpCqCqrCpCqr$ T6

T6+MM3 $\dashrightarrow CpCqCrCrs \Rightarrow CpCqCrs$ M14

T2+MM4 $\dashrightarrow CCpqr \Rightarrow Cqr$ M15

M5+M15 $\dashrightarrow CpCqr \Rightarrow CqCpCsr$ M16

* M16+M11 $\dashrightarrow CpCqr \Rightarrow CqCpr$ M17

* T1+M17 $\dashrightarrow CCpqCCrpCrq$ T7

* T7+M14 $\dashrightarrow CCpCqrCCqpCqr$ T8

* T2+M7 $\dashrightarrow Cpp$ T9

* T9+M17 $\dashrightarrow CpCCpqq$ T10

T10+MM4 $\dashrightarrow CCCpqqr \Rightarrow Cpr$ M18

T7+M18 $\dashrightarrow CpCCqCprCqr$ T11

T11+M17 $\longrightarrow CCpCqrCqCpr$ T12

T12+MM4 $\dashrightarrow CCpCqrs \Rightarrow CCqCprs$ M19

* T8+M19 $\dashrightarrow CCpCqrCCpqCpr$ T13

* T13+M17⟶CCpqCCpCqrCpr T14

* T2+M15⟶CpCqCrp T15

* T2+M2⟶CCCpqrCqr Γ16

* T16+M2⟶CCCpqrCCCspqr T17

* T2+M10⟶CCpqCΓCrq T18

L'étape la plus importante est la démonstration de M17. Elle est très différente de celle que j'avais trouvé : j'avais d'abord démontré T10 obtenu par action de M15 sur CCCpqpCCpqq qui est T4+M4. De T10, on déduit M18 et M17 était obtenu par M2+M18.

Church ne donne pas les étapes des démonstrations. Notons toutefois qu'il demande de démontrer T12 avant T7 alors que le programme utilise T7 pour démontrer T12. Il demande de démontrer CCCqrrCCCpqrr qui n'est qu'un cas particulier de T17 (p/q; q/r; r/r; s/p).

On a déjà vu que tout théorème de cette axiomatique est un théorème de l'axiomatique 2. On a utilisé ceci pour montrer que T12 et T13 étaient des théorèmes de l'axiomatique 2. On ne les avait pas démontrés directement, car l'axiomatique 2 a beaucoup plus de théorèmes intéressants puisqu'elle considère aussi la négation. Le programme avait dû s'arrêter trop tôt, faute de place en mémoire.

AXIOMATIQUE 5

Cette axiomatique, dûe à Nicod, a un seul axiome et une seule connective, le Sheffer's stroke. Je n'ai pas pu avoir de document où cette axiomatique est étudiée.

axiome ||p|qr||s|ss|tq||pt|pt T1

production p,|p|qr⇒r M1

* |p|qr⟶p⇒r MM1

* p⇒|q|rs⟶p,q⇒s MM2

T1+MM1⟶|p|qr⇒||sq||ps|ps M2

T1+M2⟶||p|q|qq|||r|stp||r|stp T2

T2+MM1⟶|p|q|qq⇒||r|stp M3

T2+M2⟶||p||q|rst|||t|u|uup||t|u|uup T3

T3+MM1⟶|p||q|rst⇒||t|u|uup M4

T1+M4⟶||||pq||rp|rp|s|ss|r|qt T4

T4+M3⟶||p|qr|||st||ts|u|uu T5

T5+MM1⟶|p|qr⇒|s|ss M5

* T5+M5⟶|p|pp T6

* T6+M2⟶||pq||qp|qp T7

* T7+MM1⟶|pq⇒|qp M6

T6+M6→↔||ppp T8

M6+MM2→↔p, |qrp⇒r M7

||pqr⟶r⇒q MM3

p⇒||qrs⟶p,s⇒r MM4

T6+MM3→↔||pq|pq⇒q M8

T7+M6→↔|||pq|pq|qp T9

T9+MM1→↔||pq|pq⇒p M9

T7+M2→↔||p|qr|||rqp||rqp T10

T10+MM1→↔|p|qr⇒||rqp M10

M10+M10→↔||pq|rs⇒||qp|sr M11

T10+M11→↔|||pqr|||qpr||qpr T11

T11+MM1→↔||pqr⇒||qpr M12

M12+MM3→↔||pqr,r⇒p M13

||pqr⟶r⇒p MM5

p⇒||qrs⟶p,s⇒q MM6

T11+M2→↔||p||qrs||||rqsp|||rqsp T12

T12+M12→↔||||pqrs||||qprs|||qprs T13

T13+MM1→↔|||pqrs⇒|||qprs M14

T6+M4→↔|||p|qr|s|ss|p|qr T14

T14+M6→↔||p|qr||p|qr|s|ss T15

T15+M11→↔|||pqr||s|ss|r pq T16

T16+M14→↔|||pqr||s|ss|r|qp T17

T17+MM1→↔|pqr⇒r|qp M15

T10+M2→↔||p||qrs|||s|rqp||s|rqp T18

T18+M12→↔||||pqrs|||r|qps||r|qps T19

T19+MM1→↔|||pqrs⇒||r|qps M16

T11+M16→↔||p|qr|||qrp||qrp T20

T20+MM1→↔|p|qr⇒||qrp M17

M17+MM6→↔|p|qr,p⇒q M18

|p|qr⟶p⇒q MM7

p⇒|q|rs⟶p,q⇒r MM8

T19+M16→↔||p|qrs||p|rqs||p|rqs T21

T21+M2→↔||p|q|rst|||q|srtp||q|srtp T22

T22+MM1→↔|p|q|rst⇒||q|srtp M19

M15+M19→↔||p|q|rst⇒|||q|srpt M20

M20+M16→↔||p|q|rst⇒|p||srqt M21

M16+M21⟷||||pqrst⇒||s||qprt M22

T17+M22⟷||p||qrs||t|tt|p||rqs T23

T23+MM1⟷|p||qrs⇒|p||rqs M23

* T7+M14⟷|||pqr||r|qp|r|qp T24

* T24+M12⟷||p|qr||p|rq|p|rq T25

T25+MM1⟷|p|qr⇒|p|rq M24

* T25+M23⟷||p|qr|||rqp|p|rq T26

* T26+M24⟷||p|qr||p|rq||rqp T27

* T6+M23⟷||pq||qp|pq T28

* T7+M23⟷||pq||pq|qp T29

* M10+M8⟷||pq|qp⇒q M25

* M10+M9⟷||pq|qp⇒p M26

Les performances du programme sont moyennes. Cette axiomatique est certainement très difficile et il aurait été intéressant d'utiliser des définitions. Malheureusement les définitions à utiliser comme :

~p=|pp

⊃pq=|p|qq

n'ont pas dans le programme d'heuristiques qui leur soient bien adaptées.

Le programme a surtout démontré des Mthéorèmes liés à la permutation des variables (M6, M10, M12, M15, M17, M24). Pour cela il a d'abord démontré des théorèmes du type |a|bb ou |a|Ab ou |a|bA (A étant T6) qui donnent naissance au Mthéorème a⇒b. A la fin il a démontré T26 qui est du type |a|bc qui donne naissance aux deux Mthéorèmes : a⇒b et a⇒c.

Remarquons que T6 et T8 correspondent à ⊃pp; M9 et M8 correspondent respectivement à ∧pq⇒p et ∧pq⇒q. Enfin il a démontré 3 Mthéorèmes fondamentaux qui complètent M1 qui était donné: M7, M13 et M18.

AXIOMATIQUE 6

Cette axiomatique, dûe à Russel, est celle des principia. Un des axiomes (T5) peut se déduire des 4 autres. Dopp donne une étude de l'axiomatique obtenue en éliminant cet axiome superflu.

Le Logic Theorist de Newell, Shaw et Simon travaille sur cette axiomatique (avec T5). Leur méthode est exposée dans "Empirical exploration of the logic theory machine. A case study in heuristic" - Proceedings of the Western Joint Computer Conference - Février 57 - p. 218. Leur programme a été décrit très en détail dans "The logic theory machine : a model heuristic program" par Stefferud - Memorandum Rand RM3731CC - Juin 63. Il y donne également les démonstrations trouvées par le programme.

Nous avons séparé les résultats qui dépendent de T5 de ceux qui peuvent se déduire des 4 autres axiomes. Nous indiquons à la suite du nom d'un résultat son numéro dans les principia.

Nous avons donné des démonstrations de conjectures. La conjecture à démontrer est placée isolée sur une ligne et est suivie d'un point d'interrogation. Dans les lignes suivantes se trouve la démonstration trouvée par le programme.

axiomes
$\supset Vppp$ T1-1.2
$V\sim pVqp$ T2
$\supset Vpq Vqp$ T3-1.4
$\supset\supset pq\supset Vrp Vrq$ T4-1.6
$\supset Vp Vqr Vq Vpr$ T5-1.5

productions
$p, \supset pq \Rightarrow q$ M1

définitions
$\supset pq = V\sim pq$ D1
$\wedge pq = \sim\supset p\sim q$ D2

Les résultats qui suivent ne dépendent pas de T5

* $\supset p1 \longrightarrow p \Rightarrow q$ MM1
* $p \Rightarrow \supset qr \longrightarrow p, q \Rightarrow r$ MM2

* T1+D1 $\dashv\vdash \supset\supset p\sim p\sim p$ T6-2.1
* T2+D1 $\dashv\vdash \supset pVqp$ T7-1.3
* T7+D1 $\dashv\vdash \supset p\supset qp$ T8-2.2
* T3+D1 $\dashv\vdash \supset\supset p\sim q\supset q\sim p$ T9-2.3
T9+MM1 $\dashv\vdash \supset p\sim q \Rightarrow \supset q\sim p$ M2
T1+M2 $\dashv\vdash \supset p\sim V\sim p\sim p$ T10
T10+D1 $\dashv\vdash \supset p\sim\supset p\sim p$ T11
* T11+D2 $\dashv\vdash \supset p\wedge pp$ T12

T4+MM1 $\dashv\vdash \supset pq \Rightarrow \supset Vrp Vrq$ M3
M3+MM2 $\dashv\vdash \supset pq, Vrp \Rightarrow Vrq$ M4
$\supset pq \longrightarrow Vrp \Rightarrow Vrq$ MM3
* T4+D1 $\dashv\vdash \supset\supset pq\supset\supset rp\supset rq$ T13-2.5
T13+MM1 $\dashv\vdash \supset pq \Rightarrow \supset\supset rp\supset rq$ M5
M5+MM2 $\dashv\vdash \supset pq, \supset rp \Rightarrow \supset rq$ M6
$\supset pq \longrightarrow \supset rp \Rightarrow \supset rq$ MM4
$\supset pq \longrightarrow \supset qr = pr$ MM5

T7+MM5 $\dashv\vdash \supset Vpqr \Rightarrow \supset qr$ M7
* T1+M7 $\dashv\vdash \supset pp$ T14-2.8
T1+MM3 $\dashv\vdash Vp Vqq \Rightarrow Vpq$ M8
* T2+M8 $\dashv\vdash V\sim pp$ T15-2.10
T3+MM1 $\dashv\vdash Vpq \Rightarrow Vqp$ M9
* T15+M9 $\dashv\vdash Vp\sim p$ T16-2.11
* T16+D1 $\dashv\vdash \supset p\sim\sim p$ T17-2.12

T3+MM4⊢⊃pVqr⇒⊃pVrq M10

* T7+M10⊢⊃pVpq T18-2.20

* T18+D1⊢⊃~p⊃pq T19-2.21

T17+MM5⊢⊃~~pq⇒⊃pq M11

* T19+M11⊢⊃p⊃~pq T20-2.24

T18+M3⊢⊃VpqVpVqr T21

T3+MM5⊢⊃Vpqr⇒⊃Vqpr M12

T21+M12⊢⊃VpqVqVpr T22

T22+MM5⊢⊃VpVqrs⇒⊃Vqps M13

* T1+M13⊢⊃VpVpqVpq T23-2.40

* T23+D1⊢⊃⊃p⊃pq⊃pq T24-2.43

* T23+M10⊢⊃VpVpqVqp T25-2.41

T3+M3⊢⊃VpVqrVpVrq T26-2.30

T17+M3⊢⊃VpqVp~~q T27

T27+M12⊢⊃VpqVq~~p T28

T28+MM1⊢Vpq⇒Vq~~p M14

T18+MM4⊢⊃pq⇒⊃pVqr M15

T7+MM4⊢⊃pq⇒⊃pVrq M16

T9+MM5⊢⊃⊃p~qr⇒⊃⊃q~pr M17

T9+MM4⊢⊃p⊃q~r⇒⊃p⊃r~q M18

T17+M5⊢⊃⊃pq⊃p~~q T29

T17+MM1⊢p⇒~~p M19

Nous avons essayé de démontrer un certain nombre de conjectures. Nous n'avons pas essayé des résultats trop faciles comme ⊃pVpp (2.7 des principia) qui n'est qu'un cas particulier de T7.

- ⊃~~pp ?

T15+M15⊢Vp~~~p T30-2.13

T30+M9⊢V~~~pp T31

* T31+D1⊢⊃~~pp T32-2.14

on en déduit

T32+MM5⊢⊃pq⇒⊃~~pq M20

T32+M5⊢⊃⊃p~~q⊃pq T33

- ⊃~∧pqV~p~q ?

T32+D1⊢⊃~~⊃pqV~pq T34

T34+D2⊢⊃~∧pqV~p~q T35-3.13

- $\supset\vee\sim p\sim q\sim\wedge pq$?

T17+D1$\rightarrow\supset\vee\sim pq\sim\sim\supset pq$ T36

T36+D2$\rightarrow\supset\vee\sim p\sim q\sim\wedge pq$ T37-3.14

- $\supset\supset pq\supset\sim q\sim p$?

* T29+M18$\rightarrow\supset\supset pq\supset\sim q\sim p$ T38-2.16

- $\supset\supset\sim p\sim q\supset qp$?

* T33+M17$\rightarrow\supset\supset\sim p\sim q\supset qp$ T39-2.17

- $\sim\wedge p\sim p$?

T17+M19$\rightarrow\sim\sim\supset p\sim\sim p$ T40

* T40+D2$\rightarrow\sim\wedge p\sim p$ T41-3.24

- $\supset\supset\sim pq\supset\sim qp$?

Le programme n'a pu arriver à le démontrer. Supposons le démontré pour la suite des démonstrations en l'appelant T42 (2.15 des principia).

T42+MM1$\rightarrow\supset\sim pq\Rightarrow\supset\sim qp$ M21

- $\supset\sim\vee pq\sim q$?

T7+M20$\rightarrow\supset\sim\sim p\vee qp$ T43

T43+M21$\rightarrow\supset\sim\vee pq\sim q$ T44-2.46

T44+D1$\rightarrow\supset\sim\supset pq\sim q$ T45

T45+MM5$\rightarrow\supset\sim pq\Rightarrow\supset\sim\supset rpq$ M22

- $\supset\sim\vee pq\sim p$?

T18+M20$\rightarrow\supset\sim\sim p\vee pq$ T46

T46+M21$\rightarrow\supset\sim\vee pq\sim p$ T47-2.45

- $\supset\sim\vee pq\vee p\sim q$?

T44+M16$\rightarrow\supset\sim\vee pq\vee r\sim q$ T48

On fait dans T48 la substitution : r/p et l'on a le résultat cherché (2.48)

- $\supset\sim\vee pq\vee\sim pq$

T47+M15$\rightarrow\supset\sim\vee pq\vee\sim pr$ T49

On fait dans T49 la substitution : r/q et l'on a le résultat cherché (2.47)

- $\supset\sim\vee pq\vee\sim p\sim q$

On fait dans T49 la substitution : r/~q et l'on a le résultat cherché (2.49)

- ⊃~⊃pq⊃~pq ?

T49+D1⊣↦⊃~⊃pq⊃~pr T50

On fait dans T50 la substitution : r/q et l'on a le résultat cherché (2.50)

- ⊃~⊃pq⊃p~q ?

T48+D1⊣↦⊃~⊃pq⊃r~q T51

On fait dans T51 la substitution : r/p et l'on a le résultat cherché (2.51)

- ⊃~⊃pq⊃qp ?

T19+M22⊣↦⊃~⊃pq⊃qr T52

On fait dans T52 la substitution : r/p et l'on a le résultat cherché (2.521)

Indiquons enfin les résultats dont la démonstration fait intervenir l'axiome T5

* T5+D1⊣↦⊃⊃p⊃qr⊃q⊃pr T53-2.4

T53+MM1⊣↦⊃p⊃qr⇒⊃q⊃pr M23

* T14+M23⊣↦⊃p⊃⊃pqq T54-2.27

* T13+M23⊣↦⊃⊃pq⊃⊃qr⊃pr T55-2.6

T55+MM1⊣↦⊃pq⇒⊃⊃qr⊃pr M24

* T8+M24⊣↦⊃⊃⊃pqr⊃qr T56

T5+MM4⊣↦⊃pVqVrs⇒⊃pVrVqs M25

T26+M25⊣↦⊃VpVqrVrVpq T57

* T57+M10⊣↦⊃VpVqrVVpqr T58-2.31

Le programme a été géné par la formulation anormale de cette axiomatique qui fait intervenir dans sa production et ses axiomes une connective qui est introduite par une définition.

CHAPITRE VII

CONCLUSION

7A - AMELIORATIONS SOUHAITABLES

a - Possibilités perceptives

L'insuffisance se fait surtout sentir dans l'utilisation des définitions et la sélection des essais. Il faudrait pouvoir reconnaitre dans une partie d'un énoncé qu'une suite de symboles peut être modifiée utilement par une production ou une définition.

La détermination de l'intérêt est en général satisfaisante sauf pour les productions à deux antécédents. Cela entraine dans certains cas l'élimination du théorème qui est à l'origine et fait échouer le programme s'il a un rôle important dans l'axiomatique. Pour les productions à un antécédent, on aurait une sélection plus fine en faisant intervenir la troisième définition de la force. Cela remédierait sans doute à la baisse des performances qui se produit quand on a démontré et gardé une centaine de productions à un antécédent.

b - Possibilités métalinguistiques

Il ne semble pas nécessaire de considérer des niveaux plus élevés. Par contre il faudrait avoir une plus grande variété de MMthéorèmes et de MMMthéorèmes pour pouvoir démontrer plus de Mthéorèmes et de MMthéorèmes, et pour traiter des axiomatiques plus variées. On a exposé en 2E une extension qui pourrait se faire en donnant au programme la possibilité de travailler de façon récursive sur les formules de démonstration ou sur les énoncés.

Il faudrait avoir aussi la possibilité de créer de nouvelles définitions. Cela ne poserait pas de difficulté à programmer. Mais il serait beaucoup plus délicat de les choisir (à partir de notions sémantiques ou de la fréquence de certaines suites de symboles).

c - Sémantique

On n'a pas introduit les méthodes basées sur les valeurs de vérité, car les démonstrations devenaient alors trop faciles. On a introduit la sémantique de façon indirecte en calculant l'intérêt des théorèmes à l'aide de celui des Mthéorèmes que l'on peut en déduire. On fait ainsi une liaison entre la langue et la Mlangue (qui est un système logique), donc on indique de façon cachée que l'axiomatique étudiée a une interprétation logique. La sémantique semble importante si l'on veut créer de nouvelles définitions : on choisit souvent un groupement de symboles parce qu'il correspond à un concept important dans une des interprétations de l'axiomatique (par exemple ≡ en logique). Elle peut guider pour poser des conjectures que l'on essayera ensuite de démontrer.

d - Heuristiques

Cela peut se faire à deux niveaux :

- améliorer les heuristiques actuelles et en particulier celles qui décident de l'application d'une définition ou qui sélectionnent les essais. Le programme fait beaucoup trop d'essais par rapport à ce que ferait un mathématicien : il profite de la rapidité de la calculatrice.

- avoir des heuristiques plus générales qui permettent de découvrir de nouvelles heuristiques. Il y a entre elles un lien analogue à celui des MMthéorèmes avec les Mthéorèmes. Cela nécessite la définition d'un language dans lequel on exprime les heuristiques. Il faudrait également donner au

programme la possibilité de manipuler ce langage. Il s'agit là d'un souhait qu'on ne sait pas encore réaliser. (Tout ce que l'on a pu faire dans cette voie est de faire dépendre les heuristiques de paramètres et de faire varier ces paramètres). Si on avait cette possibilité, on pourrait également introduire les heuristiques sous forme de données qui seraient interprétées par un programme décodant le langage dans lequel on exprime les heuristiques. D'où une beaucoup plus grande souplesse.

e - Conjectures

Il faudrait d'abord de meilleures heuristiques que celles que l'on a programmé. Avec tous les Mthéorèmes qu'il avait à sa disposition, le programme aurait dû avoir des performances bien supérieures.

Il faudrait aussi que le programme puisse se donner les conjectures qu'il va démontrer. Il semble indispensable de faire intervenir pour cela la sémantique. Un moyen détourné serait de faire conjecturer des Mthéorèmes d'après une liste de propriétés importantes donnée une fois pour toutes (commutativité, associativité, transitivité...). Il suffirait alors de savoir choisir convenablement dans cette liste.

7B - PRINCIPALES DIFFICULTES RENCONTREES

a - Absence de documentation

Bien entendu, il n'y avait rien sur le problème étudié, mais le manque de documentation s'est fait sentir aussi dans les domaines voisins et en particulier par l'absence de travaux psychologiques utilisables. L'étude psychologique de la manière dont un homme resoud un problème en est encore à ses débuts. A la rigueur on peut trouver quelques description des intermédiaires par lesquels il passe, mais il y a à peu près rien sur les mécanismes qu'il emploie. Les travaux des mathématiciens ne sont guère utiles, car ils ont l'habitude de présenter leurs découvertes sans aucun rapport avec les étapes qu'ils ont dû franchir pour faire ces découvertes.

b - Insuffisance des machines

On n'est pas tellement géné par leur lenteur. En effet, nous sommes en compétition avec le cerveau humain dont les cellules nerveuses ont des vitesses qui se chiffrent en millisecondes. Il est vrai qu'il travaille en parallèle. La principale gêne vient de la faiblesse de capacité des mémoires. On ne pouvait disposer pour la mémoire de travail que d'environ 500.000 bits, ce qui est ridicule vis à vis de la capacité du cerveau humain.

c - Difficulté du problème

Le problème en lui-même est très difficile, même pour un bon mathématicien.

Frege avait pris simultanément dans son axiomatique :

$$\supset\supset p\supset qr\supset\supset pq\supset pr$$

$$\supset p\supset qp$$

$$\supset\supset p\supset qr\supset q\supset pr$$

Il n'avait pas vu que le troisième est une conséquence des deux premiers, ce que le programme a parfaitement vu quand il a étudié l'axiomatique 1.

On peut également citer Lukasiewicz dans "Aristotle's syllogistic" - Clarendon Press :

"One must be very expert in performing such proofs if one want to deduce from the axioms T_1-T_3 the law of commutation $CCpCqrCqCpr$ or even the law of simplification $CpCqp$".

L'axiomatique à laquelle il fait allusion a été étudiée sous le nom d'axiomatique 2. Le programme a trouvé une démonstration pour ces deux résultats (pour le premier il a fallu passer par l'intermédiaire de l'axiomatique 4).

Le programme semble du reste relativement plus à l'aise dans les axiomatiques difficiles. Pour juger de la difficulté du problème, il est intéressant d'en essayer une, par exemple l'axiomatique 3.

Il faut remarquer que le mathématicien a une facilité que le programme n'a pas : les considérations qui l'amènent à choisir les axiomes lui donnent souvent en sens inverse la démonstration de résultats clés. C'est certainement ainsi que Lukasiewicz a pu obtenir la démonstration de $CCpqCCqrCpr$ dans l'axiomatique 3.

7C - BILAN DE L'ETUDE

Mon but n'était pas de démontrer de nouveaux théorèmes, mais de voir comment une machine pouvait démontrer des théorèmes. Je voulais également essayer de dégager des principes qui puissent s'appliquer à d'autres domaines de l'intelligence artificielle comme les jeux.

Deux faits se dégagent de ce travail :

- importance de l'archivage des résultats trouvés. Cet archivage doit être sélectif et structuré : on doit garder une faible partie de ce que l'on trouve, mais choisir judicieusement ce que l'on garde en tenant compte de ce que l'on a trouvé par ailleurs. Il faut aussi associer les informations de nature voisine de façon à accélérer les recherches.

Ceci doit être fait au moment de la découverte, mais il faut y revenir ensuite, s'assurer que certains résultats ne sont pas devenus inutiles, ont été mal rangés ou que les associations faites sont insuffisantes ou mauvaises.

- importance de l'utilisation des niveaux métathéoriques élevés. On a tout de suite une beaucoup plus grande généralité : les lois à ces niveaux sont assez peu dépendantes du problème étudié. De plus cela permet d'établir des métathéorèmes qui sont des groupements intéressants d'opérations élémentaires. Plutôt que d'être obligé de retrouver à chaque fois la bonne succession des opérations, on le fait une fois pour toutes et on les applique ensuite globalement.

Bien entendu, le programme n'a pas découvert de théorème nouveau. Cela est impossible dans la logique des propositions ! Mais il a très probablement trouvé des démonstrations nouvelles, dont certaines présentent une réelle difficulté : par exemple $CC{\sim}ppp$ dans l'axiomatique 1 ou $CCCpqpp$ dans l'axiomatique 3. (Il est difficile de recenser toutes les démonstrations qui ont été trouvées par les logiciens). De toutes les façons les théorèmes qu'il trouvait étaient tous nouveaux pour lui.

Tout ceci n'aura bien sûr un intérêt pratique que quand un programme opérera dans des théories mathématiques beaucoup plus complexes et cela ne se fera sans doute pas avant quelques dizaines d'années. Mais avant d'en arriver là, il est nécessaire de dégager des méthodes générales en étudiant des théories plus simples. C'était le but de cette étude.

DEUXIÈME THÈSE

Propositions données par la Faculté

"Approximation de fonctions expérimentales en présence de bruit".

Vu et approuvé :

Paris, le 4 MARS 1966

Le Doyen de la Faculté des Sciences,

Marc ZAMANSKY

Vu

et Permis d'imprimer :

Le Recteur de l'Académie
de Paris,

Jean ROCHE